NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DESIGN, IMPLEMENTATION, AND ANALYSIS OF AN ARMY INTERACTIVE MULTIMEDIA THREAT IDENTIFICATION TRAINING SYSTEM

by

Douglas S. Miller

September 1999

Thesis Advisor:	Geoffrey Xie
Thesis Co-Advisor:	John S. Falby

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 1999	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE DESIGN, IMPLEMENTATION, AND ANALYSIS OF AN ARMY INTERACTIVE MULTIMEDIA THREAT IDENTIFICATION TRAINING SYSTEM	5. FUNDING NUMBERS
6. AUTHOR(S) Miller, Douglas S.	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING / EXPERIMENTERING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING / EXPERIMENTERING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

Correctly identifying a weapon system as "friend" or "foe" is vital to the success of the Army mission. Incorrect identification can leave the enemy to fight another day or cause a fratricide event. A current threat identification training method is to use Army Field Manual (FM) 1-402, which has not been updated since 1984, and is difficult to tailor towards specific and evolving threat training and mission requirements. This thesis, therefore, has two main purposes: development of a modifiable computer-based program that individual units can easily tailor to meet their current threat training requirements, and a statistical analysis to determine if Computer-Based Training (CBT) is suitable for individual soldier threat identification training. The CBT application developed for this thesis is dynamically linked to image and text files maintained by the unit-training officer, thus allowing for modification and updates as required. Experimental results indicate that using this CBT application can be a suitable, and in some aspects better, training tool than FM 1-402. After a 20-minute study period, experiment participants using the CBT had an average nomenclature final test score increase of 38.4 percent over a similarly experienced control-group.

14. PARTICIPANTS TERMS Computer-based training	15. NUMBER OF PAGES 113
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFI- CATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

DESIGN, IMPLEMENTATION, AND ANALYSIS OF AN ARMY INTERACTIVE MULTIMEDIA THREAT IDENTIFICATION TRAINING SYSTEM

Douglas S. Miller
Captain, United States Army
B.A., University of Colorado, Boulder, 1989

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MODELING, VIRTUAL ENVIRONMENTS, AND SIMULATION

from the

NAVAL POSTGRADUATE SCHOOL

ABSTRACT

Correctly identifying a weapon system as “friend” or “foe” is vital to the success of the Army mission. Incorrect identification can leave the enemy to fight another day or cause a fratricide event. A current threat identification training method is to use Army Field Manual (FM) 1-402, which has not been updated since 1984, and is difficult to tailor towards specific and evolving threat training and mission requirements. This thesis, therefore, has two main purposes: development of a modifiable computer-based program that individual units can easily tailor to meet their current threat training requirements, and a statistical analysis to determine if Computer-Based Training (CBT) is suitable for individual soldier threat identification training. The CBT application developed for this thesis is dynamically linked to image and text files maintained by the unit-training officer, thus allowing for modification and updates as required. Experimental results indicate that using this CBT application can be a suitable, and in some aspects better, training tool than FM 1-402. After a 20-minute study period, experiment participants using the CBT had an average nomenclature final test score increase of 38.4 percent over a similarly experienced control-group.

TABLE OF CONTENTS

I. INTRODUCTION

A. AREA OF RESEARCH

The goal of this thesis is to develop a PC-based, multimedia-oriented, modifiable threat identification training system for individual threat identification training and testing tasks within Army Aviation Units. The content of this system must be easy to use and update. Training with this system should produce at least the same amount of learning and retention as training with FM 1-402.

U.S. Army units have the requirement to train and test their soldiers on threat identification according to current regulations. These units have specified regional areas of interest that they train to for contingency operations and thus have different threat equipment that their soldiers must be able to correctly identify. For example: a soldier assigned to a unit in Germany should focus on a European threat package, while a soldier assigned to Korea should concentrate on a North Korean threat package. Therefore, this thesis also stresses the modifiability of the developed system to meet training requirements for specific units within the Army, and the feasibility of maintaining different threat vehicle databases at individual units with minimum effort. The base program can be used Army wide, while each unit can tailor the trainer to meet its own specific threat identification training needs.

Macromedia's Director 6.0 and Authorware 5.0 Attain are currently available multimedia software design and Computer-Based Training (CBT) packages. Director 6.0 was used to develop the training portion of the thesis and Authorware 5.0 Attain for the testing and usability portion of the thesis.

B. RESEARCH QUESTIONS

- What are the current methods of training threat identification tasks in Army Aviation Units?
- Can a multimedia threat identification training system enhance this training and thereby increase the performance of soldiers when required to correctly identify threat vehicles in a hostile situation?
- How can Macromedia Director 6 software increase the quality of threat identification training for Army Aviation Units?
- How can the Macromedia Authorware 5.0 Attain software increase the quality of threat identification training for Army Aviation units?

C. BENEFITS OF STUDY

The current quality of threat identification training available in Army units is unacceptably low. FM 1-402 is a primary training medium that is available for threat identification training. Unit personnel use this manual because it is what is readily available to them. This manual contains photographs of low quality, and it is difficult to tailor to evolving threat requirements at the unit level. FM 1-402 has not been updated since 1984 and does not include thermal image photographs of equipment as viewed from the numerous thermal-imaging direct fire weapon systems currently in use. Because of this, we are putting our soldiers at an extreme disadvantage in future conflicts in the arena of rapidly and correctly identifying a potential adversary. Continued high levels of "fratricide" events at the Army's training centers and fratricide incidents during Operation Desert Storm are just two examples showing a need for improved threat identification training. According to the Army's Training and Doctrine Command

(TRADOC), "Fratricide is the employment of friendly weapons and munitions with the intent to kill the enemy or destroy his equipment or facilities, which results in unforeseen and unintentional death or injury to friendly personnel" [CALL92-4]. While the ultimate responsibility for firing a weapon system rests with the individual pulling the trigger, we as leaders must ensure that we provide the absolute best possible training for our soldiers.

We have made tremendous improvements utilizing computer technology in many facets of planning, training, and execution of Army missions over the last decade and a half. Unfortunately, we still train our soldiers in threat identification in virtually the same way we have for the last 10 to 15 years. Threat identification CBTs, such as the one developed for this thesis, overcome the limitations of current training methods. These systems can be updated to incorporate current threat system images and identification data, and can also be tailored to meet specific needs. The quality of threat identification training that soldiers receive will increase, thus improving their ability to correctly identify friendly and enemy weapons systems, reducing the potential for fratricide.

D. METHODOLOGY

To begin this study current Army requirements to conduct threat identification training and testing were reviewed. The requirements were integrated with the author's experience as a Platoon Leader and Company Commander in Cavalry and Attack Helicopter units. This provided a basis for development of a threat identification CBT system which focuses on the training and testing of individual threat identification skills. The Mobile Threat Identification Training System (MTTS) developed for this thesis is therefore composed of two main subsections: training, and testing.

The MTTS training subsystem was developed using Macromedia Director 6 software. An experiment that compared the training subsystem versus traditional methods of training in threat identification applications was then conducted. It soon became evident that the Director 6 package was suited for the training portion of the multimedia package, but fell short in the tools required for modifiability, testing, and record maintenance. At this point it was necessary to explore the capabilities of a dedicated CBT application. The system used for this thesis was Macromedia Authorware 5.0 Attain. This software package was used to study the modifiability and training record maintenance required for the MTTS.

E. RESEARCH OUTPUT

The output includes an executable computer-based threat identification training system that can be distributed to Army Aviation units. These units will be able to utilize this system without having either of the software development packages, Macromedia Director 6.0 and Authorware Attain 5.0, installed on their unit PC's. The software training system is provided "as is." All guarantees either expressed or implied are disclaimed as to the software and its quality, performance, or fitness for any particular use.

F. ORGANIZATION OF THE STUDY

The rest of this thesis is organized into the following chapters:

Chapter II: Background This chapter covers a brief background of CBT, military training, fratricide, and current methods of threat identification training in Army Aviation units.

Chapter III: Development of the threat identification training system This chapter covers current CBT development concepts and methodology. It then focuses on the specific requirements of the software packages (Director 6.0 and Authorware 5.0) used to develop the training system.

Chapter IV: Usability Fundamentals of system usability are covered in this chapter. Instructions for using both the training and testing portions of the MTTS are included. The issue of tailoring the MTTS to unit specific content is also addressed. This chapter concludes with a discussion of program upgrades and maintenance.

Chapter V: Statistical test of CBT versus traditional training methods This chapter covers the experiment conducted to analyze the CBT developed for the thesis versus a traditional method of training Army Aviation units in threat identification tasks.

Chapter VI: Summary and recommendations for future research This thesis concludes with a brief summary of the work conducted during the study and a list of recommendations for future research.

II. BACKGROUND

A. INTRODUCTION

This chapter covers a brief background of CBT and links CBT to the specific task of threat identification training within Army Aviation units. Section B covers an overview of CBT, and reasons for using CBT in general training environments. Section C discusses possible uses for CBT within the military and gives a brief overview of fratricide. Section D addresses current threat identification training for U.S. Army Aviators.

B. COMPUTER-BASED-TRAINING OVERVIEW

With advances in computing power over the last few years, the ability to develop and run interactive multimedia applications has advanced tremendously. Today's desktop and even laptop computers have the computational power, speed, and storage capacity to handle content-intensive multimedia software applications.

Parallel with the gains in computing technology have also come rapid advances in training mediums available to instructors. This field of newly developed training mediums has become filled with terms: computer-based training (CBT), computer-assisted instruction (CAI), online learning, multimedia instruction, and digital multimedia instruction, to name a few. Each of these terms and existing training systems put a slightly different spin on the same basic theme. Easton defined computer-based training as "the use of computers and multimedia technology for training in a way that promotes student interest and motivation"[EAST97].

Advances in technology have allowed developers to create CBT applications that include the full range of multimedia content: graphics, animation sequences, high quality digital images, sound files, and video segments. The ability to combine a wide variety of multimedia content is a great advantage to increasing retention of new knowledge. Bixler and Bergman observed that "on average, people remember: 10% of what they read; 20% of what they hear; 30% of what they see; and 50% of what they hear and see" [BIXL77]. Today's multimedia capable PCs allow developers to take advantage of this fact and "create a complete multi-sensory learning program, allowing students to interact with the material, and to learn according to their own needs, pace, and learning styles"[EAST97].

Montague studied how the Navy could use CBT systems for training and listed the following reasons for using CBT systems [MONT84]:

- Reduce the variability of instruction
- Reduce requirements for presence of instructors
- Make training available at any time to match student availability
- Provide training at remote areas where instructors are unavailable

Another factor to consider when developing and using a computer-based training application is the dynamic content of the course material. If the course material requires frequent updates to remain current or student training would benefit from frequent updates to training material content, computer-based training has an advantage over written course materials. Key changes and updates can easily be edited into CBT applications. Depending on the specific CBT utilized, developers can then post updated applications to their course website, or distribute updated versions of the software package. Users then have immediate access to the most current training information. In a rapid deployment situation for an Army unit, this type of immediate update of a specific threat's weapons systems would be extremely helpful.

Recent trends show that "while twenty-two percent of sampled companies were using multimedia training in 1996, thirty-one percent of leading edge and eighty-one percent of Benchmark Forum companies were developing training via multimedia applications. These numbers are expected to grow significantly" [SALO98]. The Army should tap into this multimedia training to the same extent as the civilian education and business worlds have.

C. CBT AND THE ARMY

1. Fratricide Prevention

Every time a soldier fires a direct fire weapon at a threat, he or she must be certain that the vehicle is indeed an enemy vehicle to avoid fratricide. The consequences of mistaken identity are immense.

In the cold war era, it was relatively easy to distinguish the friend or foe status of vehicles. A fairly clear line was established between the countries and equipment of NATO and Warsaw Pact countries. The NATO forces stationed along the East-West German border knew what forces they where up against in great detail. The equipment composition of these forces was fairly static, and personnel in the units that patrolled this border took daily tests to ensure they could correctly identify the threat equipment they faced.

Since the fall of communism, NATO has expanded its ranks to include former members of the WARSAW PACT countries. These countries still maintain the vehicles and equipment that were recently on the "wrong side" of the border. What was once clearly an enemy vehicle can now be friendly. In some cases, both friendly and enemy forces may use the same equipment. In this situation it is even more critical that soldiers know exact weapon nomenclature identification of the systems they are facing. During Operation Desert Storm, allied coalition forces faced an Iraqi Armor threat that included

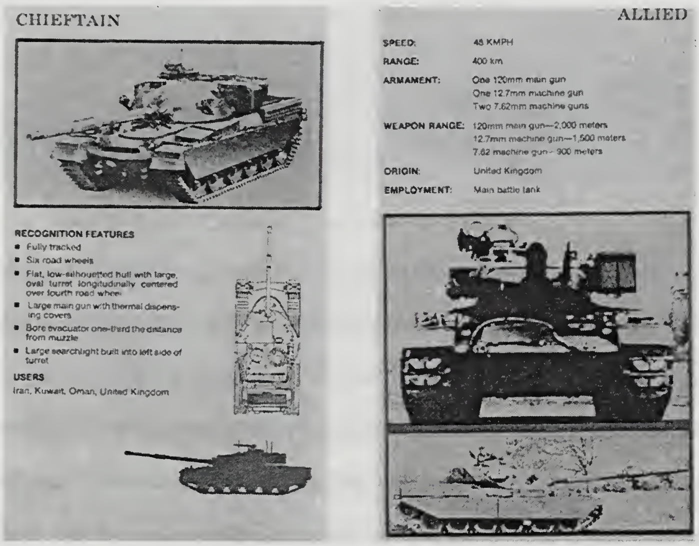
Russian built T-55 through T-72 tanks and also a variety of French and British built tanks. Current operations in the former Yugoslavia have U.S. military forces working with multinational forces and their equipment from around the world, including these same types of Russian built tanks.

With this current status of world affairs and the proliferation of weapons systems throughout the world, the friend or foe status of vehicles facing U.S. Military Forces has become clouded at best and places an even greater emphasis on correctly identifying the exact type of vehicle. This task has become more difficult as the ability to concentrate on one specific foe and their equipment has passed. It would seem then that the Army could benefit from having a training system in place that can be rapidly modified to the threat-training package its soldiers will face when deployed.

2. Linking Army Training with CBT

The Army uses a crawl, walk, run approach to training. Soldiers must first demonstrate proficiency in basic tasks prior to moving on to more complicated tasks associated with a stated learning objective. If the stated learning objective were to correctly identify armor vehicles, a soldier would first start by learning the basic daylight photographs of these types of friendly and enemy vehicles. Once a demonstrated level of proficiency has been attained at this “crawl” stage, the soldier would advance to identifying the thermal images of the same tanks. Once again, when proficiency has been demonstrated at this given level of training, the soldier would move on to a “run” training scenario. This level of training could include identifying numerous vehicles in some type of simulated battlefield training environment.

A current threat identification training method for Army Aviators is to use Army Field Manual (FM) 1-402. This manual, last updated in 1984, has three photographs of each piece of equipment and a written description of that vehicle's main recognition features (See Figure 1, below).



CHIEFTAIN

RECOGNITION FEATURES

- Fully tracked
- Six road wheels
- Flat, low-silhouetted hull with large, oval turret longitudinally centered over fourth road wheel
- Large main gun with thermal dispensing covers
- Bore evacuator one-third the distance from muzzle
- Large searchlight built into left side of turret

USERS

Iran, Kuwait, Oman, United Kingdom

ALLIED

SPEED:	48 KMPH
RANGE:	400 km
ARMAMENT:	One 120mm main gun One 12.7mm machine gun Two 7.62mm machine guns
WEAPON RANGE:	120mm main gun—2,000 meters 12.7mm machine gun—1,500 meters 7.62 machine gun– 900 meters
ORIGIN:	United Kingdom
EMPLOYMENT:	Main battle tank

Figure 1: Example Pages from FM 1-402.

Much of the manual is out of date with photographs that are not current and are of low quality. This method of training provides neither interaction nor immediate positive reinforcement in the learning process; it is basically a static learning tool. When a soldier studies the manual, he is not immediately provided the opportunity to check and enforce what he is learning. Hailey and Hailey found that "worksheets and their online equivalents force the learner to write down important points" [HAIL98], thereby increasing the retention level during a given study period. This periodic reviewing of knowledge and checking of important learning objectives is vital to the learning process.

With the current method of training available to soldiers, the best they can do is use flash cards to test if they are progressing in their learning. Figure 2 shows an example of the flash card training device currently available to soldiers.

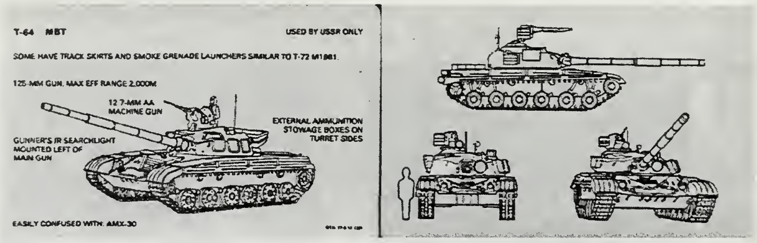

T-64 MBT
USED BY USSR ONLY
SOME HAVE TRACK SKIRTS AND SMOKE GRENADE LAUNCHERS SIMILAR TO T-72 M1981.
125-MM GUN, MAX EFF RANGE 2,000M
12 7-MM AA MACHINE GUN
EXTERNAL AMMUNITION STOWAGE BOXES ON TURRET SIDES
GUNNER'S IR SEARCHLIGHT MOUNTED LEFT OF MAIN GUN
EASILY CONFUSED WITH: AMX-30

Figure 2: Flash Card Training Device Example.

While this method of training is static and not easily modified, circumstances around the world are dynamic. A computer-based training application that incorporates audio, video, and photographs, and allows the user to interact while learning, may be the solution. Computer-based training systems can easily incorporate periodic reviews of material to reinforce critical learning objectives, and provide immediate feedback to the user. By incorporating video, graphics, and audio, such a system can fully engage the soldier in an interactive learning situation.

The Army's Center for Army Lessons Learned (CALL) Handbook 92-3 lists three primary causes of fratricide: low situational awareness of the battlefield, mistaken identification, and weapons errors [CALL92-3]. The Army has focused on situational awareness of the battlefield and thermal sight training (a sub-category of mistaken identification) to help train soldiers and reduce the likelihood of a fratricide incident. Both of these training measures are at the high end of the training spectrum.

While this is excellent training for soldiers who have progressed beyond their initial stages of threat identification training, little progress has been made in the introductory and mid-level training for new soldiers. High level complex tasks must be trained only after a solid foundation has been demonstrated in the base tasks; anything less is setting our soldiers up for failure. Here, computer-based training should be useful. By utilizing threat identification CBT systems, individual soldiers might increase the proficiency of threat identification training at all levels. The CBT system would allow soldiers to study and learn when convenient to them; developing a solid foundation in the base level recognition tasks. This would allow unit leaders to focus their efforts on planning, preparing, and executing more realistic and demanding training for their soldiers.

For CBT systems to become a viable option in threat identification training, it must be shown that they are at least as efficient in the training of threat identification tasks to soldiers as the static manual learning options currently available. Easton's definition of CBT, while adequate for general instructional purposes, needs enhancement for military CBT purposes. In addition to the Easton's definition, the CBT must also enhance the student's ability to train a specified task to standard (student in this case being a soldier). Chapter V provides the results of a study addressing this concern.

Finding time to train soldiers is difficult for commanders. Countless training distracters continually interfere with mission oriented training. This, coupled with an attitude of "do more with less", makes it extremely difficult to get large groups of soldiers together for dedicated blocks of instruction and training sessions. Even when a block of time is available for a large audience, it is often difficult to find an instructor with the suitable skills to train the group. This is another training situation in which CBT may increase the quality of available training. Kearsley developed a "CBT Benefits

Checklist" (Figure 3) to aid in assessing a training situation and the potential benefits to be derived from applying CBT technology [KEAR83].

CBT BENEFITS CHECKLIST

		Yes	No
1.	<u>Increased Control</u>		
	Are existing materials poorly used?	[ ]	[ ]
	Are existing training programs taught inconsistently?	[ ]	[ ]
	Is standardization of training important?	[ ]	[ ]
	Is detailed tracking of learning needed?	[ ]	[ ]
2.	<u>Reduced Resource Requirements</u>		
	Is decentralized training possible?	[ ]	[ ]
	Is higher student throughput desired?	[ ]	[ ]
	Is a higher student-to-instructor ratio desired?	[ ]	[ ]
	Is expensive equipment needed for training?	[ ]	[ ]
3.	<u>Individualization</u>		
	Is there considerable variation in student background?	[ ]	[ ]
	Is there considerable variation in student abilities?	[ ]	[ ]
	Is there likely to be considerable student variation in terms of learning progress?	[ ]	[ ]
	Does the instruction have to stand alone (e.g., self-study)?	[ ]	[ ]
4.	<u>Timeliness and Availability</u>		
	Is it necessary to provide training to many students as quickly as possible?	[ ]	[ ]
	Is it desirable to provide training on demand?	[ ]	[ ]
	Is there a problem with students forgetting due to premature training?	[ ]	[ ]
	Is there a problem with a shortage of qualified instructors?	[ ]	[ ]
5.	<u>Reduced training Time</u>		
	Would time savings in training be worthwhile?	[ ]	[ ]
	Can the training system and organization be changed to capitalize on time savings?	[ ]	[ ]
6.	<u>Improved Job Performance</u>		
	Is the quality of job performance a critical training concern?	[ ]	[ ]
	Are there job performance problems that improved training can address?	[ ]	[ ]

		Yes	No
7.	Convenience		
	Do employees already use a computer system for their jobs?	[ ]	[ ]
	Could CBT be integrated into existing jobs or equipment?	[ ]	[ ]
8.	Change Agent		
	Is there a need for new training approaches or methods?	[ ]	[ ]
	Could CBT lead to improved personal or organizational productivity?	[ ]	[ ]
9.	Increased Learning Satisfaction		
	Is the attrition or failure rate high?	[ ]	[ ]
	Is there a problem with student motivation?	[ ]	[ ]
10.	Reduced Development Time		
	Is the large-scale development of training materials and programs involved?	[ ]	[ ]
	Is immediate revision/update of training material important?	[ ]	[ ]
	Is instruction developed in terms of competency-based framework?	[ ]	[ ]

Figure 3: CBT Benefits Checklist [KEAR83].

Kearsley's checklist is not a conclusive evaluation tool. It is meant more as a guideline to decision-makers as to when their particular training situation might benefit from using CBT. Based on Kearsley's checklist, Montague's reasons for using CBT, and the particular requirements of the Army, it appears that the use of CBT could help increase the quality of threat identification training.

D. ARMY AVIATION THREAT IDENTIFICATION TRAINING

1. Initial Training

Newly assessed U.S. Army Aviation Officers and Warrant Officers attend Initial Entry Rotary Wing Training (IERW) at Fort Rucker, Alabama. Initial flight training is broken up into distinct phases. At the end of each phase, a student must pass an end-of-phase check-ride with an Instructor Pilot (IP) before progressing to the next phase. A check-ride consists of both an oral portion, and a hands-on flight evaluation portion. The basic combat skills (BCS) phase is the first time students are held responsible for threat identification tasks. The roughly forty students per class receive approximately ten hours of academic instruction as a group in a classroom format. Students also receive individual instruction of a varying amount from their IP. Students must pass threat identification exams during this phase of flight school to continue in the program. Exams take place in both the large classroom format and individually with IPs during the BCS end-of-phase flight check-ride. Upon completion of IERW, pilots are assigned to an operational Army unit for flight duties.

2. Unit Sustainment Training

The training level of an aviator is broken into three readiness level (RL) categories. RL3 is the lowest flight readiness level and RL1 is the highest. Aviators are placed or "designated" into one of these RLs based on their previous experience and demonstrated proficiency. A pilot's RL roughly equates to their ability to accomplish the unit's designated missions. Aviators that come straight from IERW are all placed in an RL3 status and within a designated time they must progress through RL2 to RL1. To progress to the next RL, an aviator must demonstrate proficiency during a check-ride in designated tasks to a unit IP. Tasks range from basic flight skills to complex mission related tasks required for the unit to accomplish its wartime mission. Once an aviator is designated as RL1, they must pass an annual check ride to maintain their RL designation.

The manual that describes the task, condition, and standard for all flight-related tasks is the Aircrew Training Manual (ATM).[1]

Army Training Circular (TC) 1-209 is the ATM for the OH-58D Helicopter.[2] Identification of major U.S., allied, or threat equipment is listed as a base task for all aviators to perform prior to any progression in RL status. The task, condition, and standard for this training scenario are given below.

TASK: Identify major U.S. or allied equipment and major threat equipment.

CONDITIONS: In a tactical or classroom environment.

STANDARDS:

1. Without the use of references, correctly identify major U.S. or allied equipment expected to be in the area of operations.

2. Without the use of references, correctly identify major threat equipment expected to be in the area of operations by NATO nomenclature per FM 1-402.

DESCRIPTION: Identify U.S., allied, or threat equipment expected to be in the area of operations while looking at the actual equipment or when shown photographs or mock-ups of the equipment.

An additional document that units reference for training is their Tactical Standard Operating Procedures (TACSOP). This unit-generated document generally covers two critical items: how individual Army units accomplish their Mission Essential Task List

---

[1] Each type of Army aircraft has its own specific ATM.
[2] The manual references FM 1-402 for vehicle descriptions and the unit standard operating procedure on training and testing specifics.

(METL)[3], and the training they are required to conduct on a periodic schedule. For instance, an Air Cavalry Unit TACSOP would have specific guidance on how that unit conducts a reconnaissance mission, and how often they must train this task to maintain their proficiency. Included in many TACSOPs is a listing of required friendly and enemy vehicles that personnel are required to know. These are the vehicles that unit personnel study and unit IPs test on the check-ride.

E. ROC-V

ROC-V is a threat identification training system that was developed by the Army's Night Vision and Electronic Sensors Directorate. This system is primarily geared towards training ground (M1 Tank, M2/M3 Bradley Fighting Vehicle, and LAV-25) weapon systems crews in thermal vehicle threat identification. ROC-V comes with a pre-configured set of 38 different weapons systems, both enemy and friendly, that users can train and test themselves on. It also contains video of weapons systems, and allows users to view systems at different angles and distances. The ROC-V system includes a feature that allows unit trainers to select which of the 38 pre-included vehicles to train and test their personnel on. The ROC-V system is not further modifiable at the unit level. Modifiability at the unit level is a feature that would greatly increase the value of a threat identification CBT for the various units within the U.S. Army.

F. SUMMARY

This chapter has given the reader an overview of CBT, military training, and threat identification training in Army Aviation units. It addressed the possibility of using CBT to aid in threat identification within Army Aviation units, and briefly discussed a

[3] The unit METL is a document listing the operational tasks that the unit must accomplish to succeed in combat. These tasks receive a priority for all training resources.

currently available threat identification CBT. The development of a threat identification CBT is addressed in the following chapter.

III. DEVELOPMENT OF THE MTTS

A. INTRODUCTION

This chapter discusses the general CBT development process utilizing the five-phase Instructional System Development (ISD) process [KEAR84]. It then discusses how this process was applied to develop the MTTS for this thesis. A discussion of key features of both Macromedia software packages (Director 6.0 and Authorware 5.0 Attain) that were used to develop the MTTS is then provided. The training and testing portions of the MTTS are covered separately, with each section covering a software overview, implementation details, and limitations of each software package discovered during the design and development phases.

B. CBT DEVELOPMENT CONCEPT

A myth in today's computer focused society is that if training is presented on a computer, it must be good [KEAR83]. CBT is not a "cure all" problem solver that can be implemented without quality starting materials and a well thought out development and presentation strategy. CBT must be developed with a specific purpose, based on quality course content, and focused on a well-defined and achievable training objective. Unfortunately, this is not always the case. Kersley has observed that taking a bad training course or materials and implementing them on a computer still results in bad training [KEAR83]. A powerful solution to overcome this potential pitfall is to link a subject matter expert with an authoring expert to develop the end CBT product. This pairing is well facilitated when the ISD approach to CBT development is utilized.

ISD is a performance-oriented philosophy that refers to a set of procedures for developing instructional programs in a consistent and reliable manner [KEAR84]. The

ISD focuses on the identification of specific training requirements, derives instructional content from an analysis of mission oriented tasks, and attempts to define the best strategy, media, and presentation order for the specified training objective. The five phases of ISD are: **analysis**, **design**, **development**, **implementation**, and **evaluation**[1] (See Figure 4).

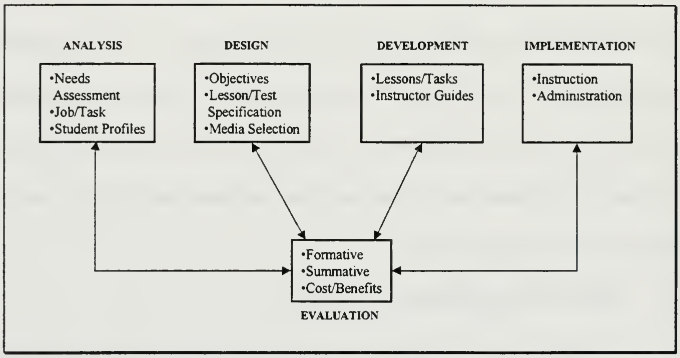


Figure 4: Five-Phases of Instructional System Development [KEAR84].

C. APPLYING ISD TO THE THREAT IDENTIFICATION CBT

1. Analysis

The focus of the analysis phase is to correctly identify an existing performance problem requiring training or determine a new training requirement. Once the specific training requirement is determined, the nature and scope of training involved must be identified. Questions to ask should include how often is this task trained, how often is this task performed, materials needed to do this task, and consequences of incorrectly

[1] It is important to note that the evaluation phase is an ongoing process linked to all phases of the process.

performing the task to be trained. Available resources, time and money constraints associated with the training program must also be identified and addressed.

According to information from the Army's Joint Readiness Training Center (JRTC)[2], Aviation units commit fratricides on an average of three incidents per rotation [CTCT97-6]. Many of these incidents occurred because some personnel failed to properly identify friendly vehicles. Correct identification of friendly and enemy vehicles is therefore the specific training task focused on for the analysis phase of the ISD.

The scope of the MTTS is the individual task performance of threat identification as outlined in the ATM (Chapter II) in a stand alone CBT. To accomplish this training objective the MTTS is further subdivided into a training portion and a testing portion. The concept of the training portion is to combine the text based information from FM 1-402 with digital photographs, video clips, and audio descriptions of key recognition features of the individual pieces of equipment and vehicles. The concept for the testing portion is to allow individual units maximum flexibility in what equipment is tested.

Available resources for development of the system were Macromedia's Director 6.0 and Authorware Attain 5.0. Adobe PhotoShop was used for digital photo editing.

2. Design

a. Human Computer Interaction

In the design phase, it is important to remember that there is not one right answer when it comes to what the final product should look like, and how it should behave [HIXD93]. The bottom line is that there should be the right amount of content

[2] JRTC is one of three Army capstone-training facilities where brigade sized units train against a dedicated opposing force (OPFOR). Units spend approximately 10 days in a battlefield environment utilizing the MILES laser detection system to monitor direct fire weapons results.

with the right amount of multimedia data presented to the user in an easy to use and consistent manner. What is intuitive to the designer, is not necessarily so for the user. A continuous effort must be made to ensure the user does not have to climb a steep learning curve to adapt to the interface [HIXD93]. While there is no one right answer in CBT design, a well thought out and customized style guide will help ensure a quality end product that is intuitive and easy is to use. Two of the more important factors to consider are color scheme and consistency.

Color is perhaps the single most overused feature in user interaction designs. Designers should limit themselves to no more than four different colors on a single screen and no more than seven different colors throughout a single application [HIXD93]. The color scheme should not cause undue eyestrain on the user. It is generally believed that viewing a bright background for long periods of time is a cause of eyestrain. High contrast between background and text colors must also be a factor when choosing the color scheme. With these considerations in mind, three main background and text color schemes were considered for the training portion of the MTTS: black background with white text, blue background with yellow text, and green background with yellow text. In the early development stage, ten individuals were shown an example of how the trainer would appear with the above combinations. Seven out of the ten individuals preferred the green background with yellow text. This color scheme therefore, was chosen for the training portion of the MTTS.

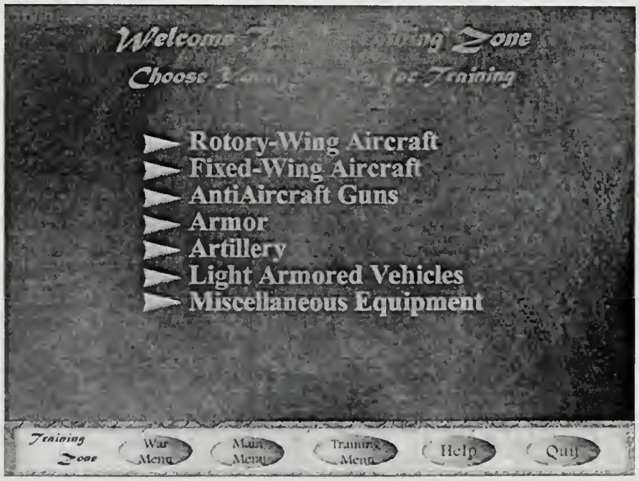
Hix recommends the principle of least astonishment when developing a CBT [HIXD93]. If something is done a certain way in an interface, users expect the same thing to be done the same way throughout the rest of the interface. Static objects such as buttons, words, and icons that appear on many screens should always appear in exactly the same location on all screens [HIXD93]. The navigation features of the training portion of the MTTS are consistent throughout the program. All main navigation features are either displayed on a tool bar at the bottom of the display window, or next to the vehicle page they take the user to. All navigation buttons are colored the same, change to the same color when the mouse is over them, and change to the same color when they are clicked. Navigation toolbars and buttons remain the same whether the user is viewing the T-80 tank information page, or the OH-58D helicopter information page. All digital images were roughly the same display size and centered in the viewing display. The test group participants of the experiment covered in Chapter V confirmed the value of this consistency. All participants rapidly gained a working understanding of the system in the allotted five-minute familiarization portion of the experiment.

b. System Flow

In the initial design phase of the system it was thought that a distinct separation would be necessary between the training and testing portions of the overall system. The concept was for the user to focus on the "training" portion while the training officer would focus on developing and administering the required tests. However, it became evident during development that the "testing" portion could have great value to the user when used as a reinforcement tool. According to Clark, because the limited capacity of working memory is rapidly overwhelmed when lots of new information is presented, it is crucial to provide frequent opportunities to use the information in working memory [CLAR98]. Designers should allow the user plenty of opportunity to clear working memory by encouraging frequent rehearsal, which moves information into long-term memory [CLAR98].

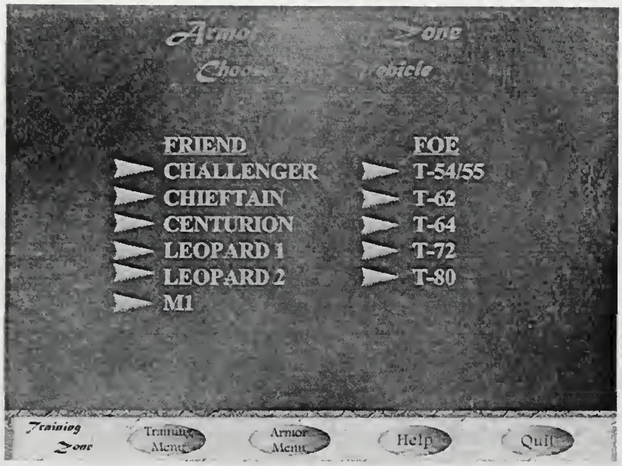
With this in mind, the author feels that using both portions of the system as a training system will provide the maximum potential for learning and retention for the user. Users can develop their own practice tests to immediately reinforce the learning process for the vehicles they are studying. Once the user is ready, he or she can take the graded test administered by the training officer.

The basic navigation of the training system is from the main home page of the system, to a specific vehicle category home page, to a specific vehicle home page, and then down to the vehicle information pages (See Figures 5 through 8).



- Entry into training system
- Choose category of weapon to train

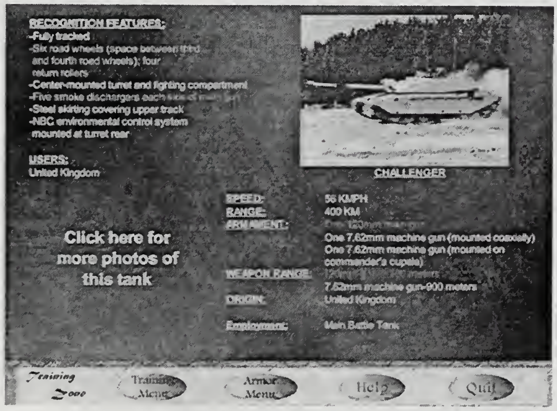
Figure 5: Main Menu.



FRIEND	FOE
CHALLENGER	T-54/55
CHIEFTAIN	T-62
CENTURION	T-64
LEOPARD 1	T-72
LEOPARD 2	T-80
M1	

- Friend and enemy listing of vehicles In the category chosen

Figure 6: Vehicle Category Main Page.



RECOGNITION FEATURES:
-Fully tracked
-Six road wheels (space between third and fourth road wheels); four return rollers
-Center-mounted turret and fighting compartment
-Five smoke dischargers each side of main gun
-Steel skirting covering upper track
-NBC environmental control system mounted at turret rear

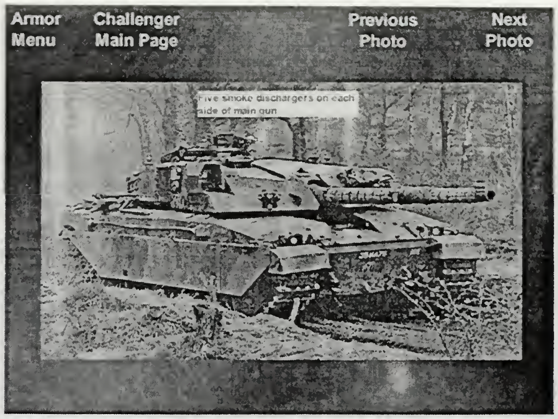
USERS:
United Kingdom

CHALLENGER

SPEED:	56 KMPH
RANGE:	400 KM
ARMAMENT:	One 120mm main gun
	One 7.62mm machine gun (mounted coaxially)
	One 7.62mm machine gun (mounted on commander's cupola)
WEAPON RANGE:	120mm [illegible] meters
	7.62mm machine gun-900 meters
ORIGIN:	United Kingdom
Employment:	Main Battle Tank

- Recognition features
- Weapons data
- Basic vehicle photo

Figure 7: Vehicle Home Page.



- Key recognition features
- 3-5 digital images per vehicle

Figure 8: Vehicle Specific Photographs.

The training system was limited to the Armor vehicle selection out of FM1-402 due to time restrictions in the academic environment. Once the framework for the system is in place and template is established, integrating other vehicles is an easily accomplished task (See Chapter IV for more on this subject).

The testing portion of the system has three main sections. Section One is a log in screen where the user enters personal information and chooses a test to take. Section two is the actual testing portion where the vehicles are displayed and the user types in answers. The last section is the grading and output section; the computer grades the answers of the user, displays the overall test score on the screen, and writes the test result data to a text file for inclusion in the user's training records.

c. System Considerations

Although most company-sized Army units have a personal computer as part of their authorized equipment, the quality and speed of these systems varies widely from unit to unit. The following is considered to be the minimum system configuration to use the CBT:

- 486 compatible CPU
- CD-ROM device
- At least 80 Mb free hard disk space (only if user wants to install executable files to the local hard drive)[3]
- A video card and a monitor that can achieve a resolution of at least 1024 x 768 pixels with a color palette setting of high color (16 bit)
- Multimedia capable sound system with speakers
- Mouse

Given the above system requirements are met, the system should perform with no noticeable performance lag in graphics display. If the minimum monitor resolution is not met, the digital photographs displayed on he screen will become unusable.

3. Development

a. Authoring Tools

Macromedia's Director 6.0 and Authorware Attain 5.0 were chosen as the authoring tools for developing the training and testing portions of the CBT respectively. Both of these systems are well suited for their respective rolls in CBT development. They are covered in greater detail later in this chapter.

[3] The final file size will depend on the total number and size of the digital photos sent with the system and the content covered in the training system.

b. Display Size

Decisions regarding content of graphics images, sound files, and video were determined in this phase. Display size choices in the Director platform include 640 x 480, 800 x 600, and 1024 x 768. The output display size of the final end product was set to 800 x 600 pixels, as this setting seemed to offer the best compromise between viewing screen area and end user system capabilities. The Authorware 5.0 display screen can be set to a variable setting or to any of the same settings as allowed in the Director 6.0 software. The graphical nature of both authoring platforms allow for wide latitude in the design process for size and placement of content on the viewing screen, however, once a screen size is set, it requires major redesign to the entire system if the size requirement changes.

c. Audio Files

Audio in the training system includes voice descriptions of key recognition features, and limited music during the introduction. The WAVE (.wav) audio file format was used for all sound files. Director allows for either linking the .wav files to the appropriate location in the program or importing the sound files directly into the program. Sound can add greatly to the quality of the CBT experience, but due to their size, sound files can dramatically affect the final file size of the product. Sound files for the CBT were therefore kept short in length, and not recorded at the highest quality available. The voice sound files used to identify key recognition features were generally less than five seconds in length. They were recorded at a sampling rate of two kHz, with an eight bit mono setting using the sound recorder tool that comes with Microsoft Windows 95. With the above recording options, the size of the voice recognition feature files averaged 140 KB. Since the final delivery method of the end product is on CD-ROM, there is plenty of space for these types of files to be included.

d. Digital Images

The Joint Photographic Expert Group (JPEG) format was used for all digital photographs. Adobe Photoshop 5.0 was used for size manipulation and other editing tasks required prior to using the JPEG files with the MTTS. The main considerations addressed in this stage were whether to link or import the digital photos into the application executable file, and the final display size of the digital images to the end user.

Authorware 5.0 allows the developer to either link or import a digital photo for use within the running application. The main difference between these two options is whether the JPEG file is included in the final executable file the user runs, or it remains a separate entity that must be shipped with the final product and imported at run time.

In the Director 6.0 training portion, the digital files were imported into the application and did not remain separate entities. In the testing portion, all digital images were linked to the individual photos.

When the Link To File option is selected during an import, Authorware displays the contents of the file in the Presentation window as if you imported it directly. However, the actual file isn't imported. It remains on the hard drive and Authorware creates a link to it using the path where it's located [ROBE97]. This is the key feature that allows users to modify the testing application. More specific implementation details are discussed further in the section on testing portion specifics later in this chapter.

The display size of the total product application screen limited the size that was allowed for individual images of vehicles. Within the training portion, all photos were imported as cast members with the image size ranging from 340 x 250 up to 640 x 480 pixels. The smaller file size images were used for the individual vehicle home page initial photos (see Figure 6). The smaller size was required to allow the text information for the individual vehicle image to fit on this page as well. The larger size digital images were used on the subsequent vehicle specific pages (see Figure 7). This had a large impact on the final size of the executable file. Most of the JPEG images prior to importing were less than 300KB. Once imported into the application as cast members, these files increased in size to roughly 1MB.

e. Video Files

During the initial development of the system, no video was included. This is one area for future work that could improve the quality of the overall system. This is addressed further in Chapter VI.

f. File Output

The general concept for the output text file from the testing portion of the system was designed in this phase. Test answer information is displayed at the end of a testing sequence for the user to see. This information is also written to a text file for record keeping and inspection purposes. The information included in the text file is user name, test date, test version, test score, and specific test questions missed. Since there is little that can be done with regards to incorrect spelling of a vehicle name during the testing and grading portion, both the answer key, and user answer are written to this file as well. This allows the unit-training officer to adjust scores based on miss-spellings or other unforeseen circumstances. There is no current file output for the training subsystem portion of the MTTS.

4. Implementation

The main concern in the implementation stage was that the end product be easy to use and implement in operational Army units. The person most likely in charge of this training would be the unit S-2, possibly with unit IPs acting as the primary agent for the respective flight companies. If the system does not provide a service that is of use, with minimal set up time, it will not be used.

Primary delivery means is via CD-ROM. The end product executable files can either be run from the CD-ROM itself or installed onto the computer's hard drive. Approximately 20 JPEG files for each tested vehicle were shipped with the original system. Unit training officers can add or delete from these as they deem necessary, and as operational requirements change. From this choice of images, the training officer must accomplish three tasks. First, he must select what vehicles they want to test. Second, he must copy the appropriate images into a separate folder. These images must be named `image01.jpg` through `image30.jpg`. These are the linked names that the testing package is looking for when it runs. Third, an answer sheet must be written and stored as a text file. This text file is placed in the same directory as the digital images for the test. Authorware looks for this text file to perform the grading of the user answers. This is discussed further in Chapter IV and detailed instructions for the user are included in Appendix C. Instructions are also shipped with the product as a read-me file.

5. Evaluation

As previously mentioned, the evaluation process is an ongoing and continual phase of the ISD process. As such, minor adjustments and modifications were incorporated into the MTTS program during all of the above phases. These modifications were generally the result of small group feedback provided to the author during the various stages of development.

Specific user evaluation of the training system was performed during the experiment covered in Chapter V. All participants that used the CBT were asked to provide feedback as to the usability of the system. They were also asked to list any major complaints or recommendations for improvement. Based on initial feedback, the most important thing to add to the system is the ability for users to compare different vehicles, i.e. a page that allows a user to display a photo of a T-80 tank side by side with a T-72 tank. This would allow for a greater ability to focus on differences between easily confused vehicles (See section D of Chapter VI for more information on future work).

The evaluation process continued with the product being tested with numerous IPs at Fort Rucker, Alabama. The results of this evaluation are covered in Chapter IV. Based on feedback received and available time, content was modified and the first operational version of the program was sent to operational Army Aviation units. The author hopes that feedback from these units will be incorporated in future versions of the MTTS, thus making it a better product that fits the training needs for all users.

The MTTS developed for this thesis represents the initial efforts towards developing a fully functional threat identification CBT that is modifiable and easy to use. The software systems used to develop the training and testing sections of the MTTS supported the goal of modifiability to varying levels. The training system was developed as the primary means to test the value of using CBT for threat identification training. As such, the idea of modifiability was not directly built into that portion of the MTTS. Director 6.0 does have the ability to track and update linked media into the final executable file run by the user. The Authorware software package, however, does a much better job of providing this functionality. The concept of modifiability became the main area of focus when developing the testing portion of the MTTS. The author feels that the training portion of the MTTS can become more portable with future work. Possibilities for this include developing a training system "shell" that would allow the user to develop their own individual training packages, or creating pre-packaged individual threat information objects that can be imported into the main executable file. These concepts are discussed further in Chapter VI.

D. MTTS TRAINING PORTION SPECIFICS

1. Software Overview

Macromedia Director 6.0 is a timeline-based multimedia-authoring tool. Director's easy-to-use interface lets you combine graphics, sound, video, and other media in any sequence and then add interactive features with LINGO, the program's powerful scripting language [ELLE97].

Developing products in the Director 6.0 authoring environment is fairly straightforward. The development environment is based on the metaphor of a theater production [ELLE97]. The basic components utilized in the development process are the stage, cast, and score. The final product executable file is called a movie.

The stage is where all the action of the product takes place. This is set up to mirror what the end user will see. During the development process, the author can quickly and easily see how his work will appear to the end user. Figures 4 through 7 are all examples of what the stage appears like during the authoring process. This is also what the end users see when the executable file is run.

The cast is the collective name for all individual files used in the movie. To use a specific file media (.jpg, .wav, etc.) in a Director program, an author must first import that file into the cast (See Figure 9) as a cast member. Cast members can be given descriptive names to aid the author. Once the desired file is included as a cast member, it can then be placed onto the stage. A sprite is the generic name given to the representation of an individual cast member that has been placed on the stage.

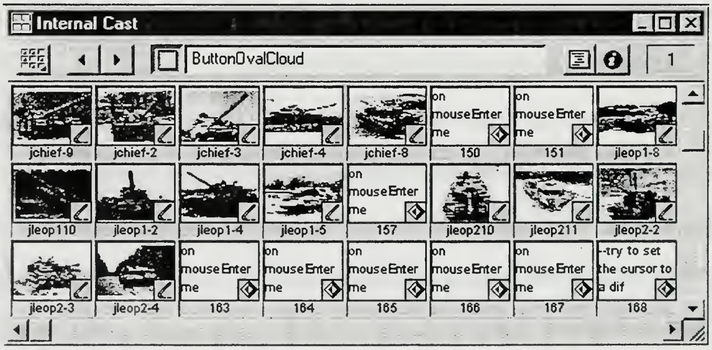


Figure 9: Internal Cast Showing Cast Members Ready for use in the Score.

The score is the visual depiction of the timeline and controls when a cast member is present on the stage. The animation, display qualities, and stage location are some of the attributes that are set in the score (See Figure 10). In Figure 10 the `jleop1-4` cast member is displayed on the stage anytime the playback head is in frames 30 through 40.

This is one location that the scripting language (LINGO) can be applied. This is covered in greater detail in section 2 b, Navigation.

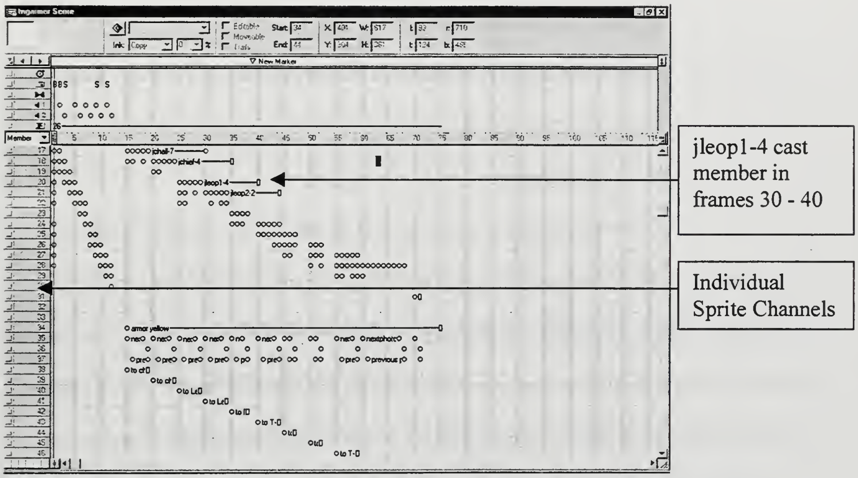

Score
New Marker
jleop1-4
jleop1-4 cast member in frames 30 - 40
Individual Sprite Channels

Figure 10: Score Example.

2. Implementation Details

a. Movie Development

Director 6.0 is a timeline based authoring tool. As the playback head progresses from the start frame to the end frame, the sprites assigned to that frame in the score are displayed on the stage. The individual sprite channels in the score can be thought of as layers. As a general rule, the cast member that is assigned to a lower number sprite channel in the score is displayed first, followed by subsequent channel number sprites. The general sequence for developing a Director movie is:

- Import as a cast member the specific file you intend to use, i.e. a specific graphic, sound file, or video clip.

- Place the cast member on the stage where you want it to appear in the final product (this instance of a cast member is called a sprite).
- Control how long the sprite is visible by assigning the appropriate length to that sprite in the score.
- Control the display properties of the sprite through the use of built in functions or through the use of the scripting language, LINGO.

b. Navigation

The base navigation mode for a Director file is to play from start to finish. Navigation can be introduced to the movie through the use of LINGO scripting commands. Depending on the functionality desired, a LINGO script in Director can be applied to either individual cast members, a sprite, or to the movie as a whole. The LINGO code itself is the same regardless of which Director element it is applied to. To achieve the navigation necessary for the trainer, a movie script was written and applied to the entire score of the movie. This movie script tells the playback head to stay in the current frame until one of the navigation buttons is activated, in this case by a single mouse click on that button. This type of script is called a score script and in this case, it is applied or active for the entire length of the movie (See Figure 11). Sprite scripts were then applied to the various navigation buttons. These scripts tell the playback head to go to the desired frame in the movie (See Figure 12).

```
on exitFrame
  go to the frame
end
```

Figure 11: Score Script Lingo Code.

```
on mouseUp me
  go to frame 8
end
```

Figure 12: Sprite Script Lingo Code.

c. Packaging for Distribution

Producing an executable file for distribution is done by creating a projector file. This is a selection option off of the file menu. Choosing this option creates a fully self-contained executable file that can be distributed via CD-ROM. The distributed executable file can either be run directly from the CD-ROM, or stored onto the users hard-drive.

3. Limitations

Due to the text-based nature of the information in the training portion, content updates will be hard to achieve without creating an updated version of the entire movie and then distributing this as an executable. Individual units will not be able to tailor the training portion to their individual unit needs. The creation of textual description information regarding specific vehicles is time consuming, and would be too much to ask the user to accomplish. It is imperative therefore that the system be designed to overcome this. It must account for a thorough coverage of weapons systems in the overall training package and allow the end user to choose what vehicles they need to train. This system can still be updated more frequently and with a more specific target audience than the traditional FM training methods. Once the basic navigation framework is laid out in the authoring tool, it is fairly easy to cut and paste new material into the package. This allows for ease of updates and tailoring specific packages to specific areas.

E. MTTS TESTING PORTION SPECIFICS

1. Software Overview

Macromedia Authorware Attain 5.0 is the flowline based authoring tool used to develop the testing portion of the system. There are three main components that a developer works with when using Authorware: icons, the flowline in the design window, and the presentation window.

The basic building blocks of an Authorware piece are icons. There are thirteen separate icon types available to the designer of an Authorware piece. Icons range in functionality from basic display icons, which can be used to display graphics and digital images, to complicated calculation icons, which are assigned various functions and variables. To use a particular icon, it is dragged from the icon palette and placed on the design window's flowline. Order of the icons on the flowline determines the order in which the functionality associated with that icon is presented to the end user. After placing an icon on the flowline, the designer assigns the required functionality to that icon and positions the associated output on the display window. The display window output is what the end user will see when running the program. After you create your flowline structure in Authorware and assign the icons different properties and options, what you end up with isn't just a representation of the piece but the source code itself, which is packaged and delivered to the end user [ROBE97].

2. Implementation Details

a. Testing Program Structure

The testing program consists of three main sections: user input, testing, and grading (See Figure 13). Specific program functions and tasks are divided amongst these three sections and are discussed in general terms below. Specific details of how Authorware functionality achieved the desired program results are covered in more detail in subsequent sections.

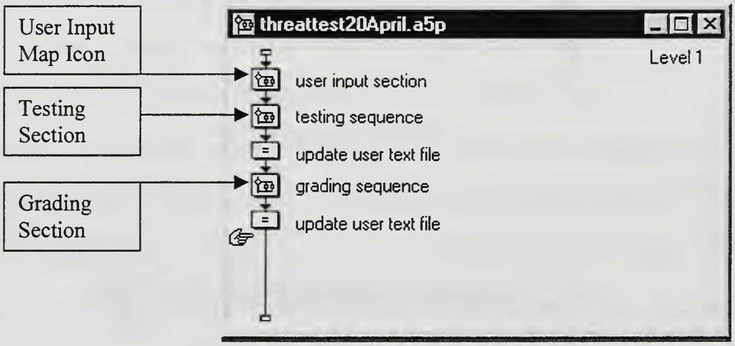


Figure 13: Authorware Main Flowline.

The user-input section displays the welcome screen and controls the logic for the login procedures (See Figure 14). The system defined variable `EntryText` is used to capture input and assign values to the user-defined custom variables of rank, `firstName`, and `lastName`. The `lastName` variable is used to define the .txt file name in which user records are stored. File I/O is discussed in greater detail in section f. After user identification information is entered, the user enters the desired test version.

EntryText again captures this input which is transferred to the appropriate custom variable.

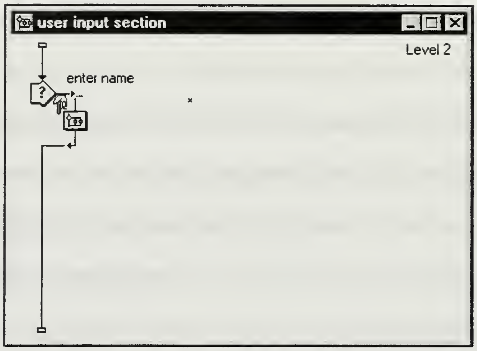


Figure 14: Contents of Map Icon for User Input Section.

This variable is then used by the system to find the answer sheet and digital images for that test version. For example, if the user enters a test version called test1, the program would look for a directory of the same name. Once this directory has been located, it would then look for a `test1.txt` file for the answer sheet. Also in this directory would be the 30 .jpg files named `image01.jpg` through `image30.jpg`.

The testing portion contains the logic to display all `image#.jpg` images, and capture and store user responses to displayed images (See Figure 15). Images are displayed for a time limit, which is set by the user. While the image is being displayed, a text entry box is also on screen. It is in this box that the user types in the answer for the displayed image. Once the answer is typed in, the user can either wait for the system clock to time out or hit the enter key to advance to the next image. For this project, a separate executable file was created for test versions that display 10, 20, and 30 images. Depending on the test version desired, the user clicks on the desired executable file. In future versions, these files could be included in one main program with the number of test questions set to a user defined variable in the login sequence.

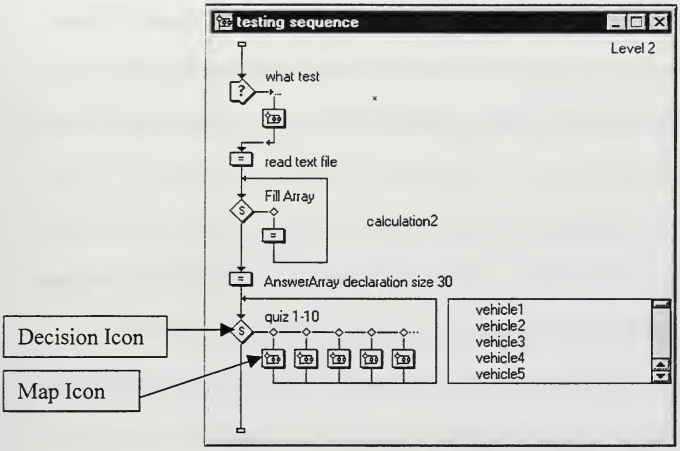


Figure 15: Testing Section.

The grading section uses a calculation icon to compare the string value elements of the `answerArray2` and the `userAnswer` arrays (See Figure 16). It then

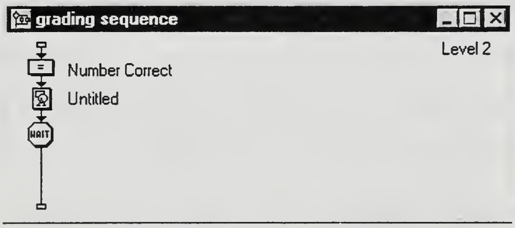


Figure 16: Grading Section.

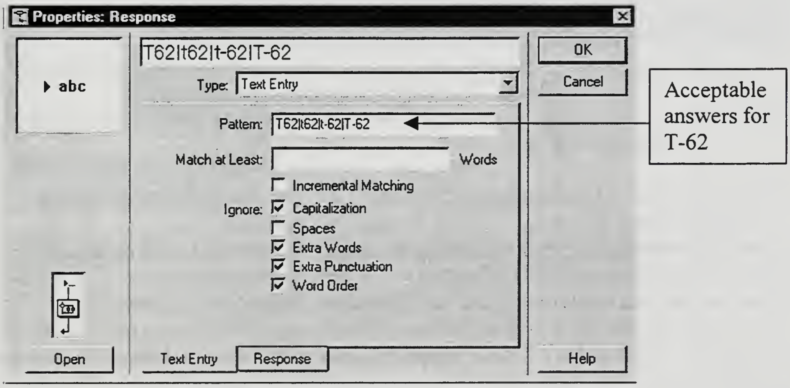
uses file output functions to write information to the user's text file. The text file contains user data, a listing of questions missed during the test, and a comparison of the user defined answer and the correct answer. This comparison and the grading is based on the answer key text file developed by the system administrator. The system administrator is defined to be the unit Intelligence Officer (S-2), unit instructor pilot, or training officer responsible for threat identification training. After running a test, the administrator can then go back and determine what vehicles need study emphasis for unit personnel. This can also be used to determine if wrong answers where a result of a miss-spelled answer or keyboard errors of the user.

b. Flowline and Icons

The icons available in Authorware allow for great flexibility in program development. Specific icons used that where valuable in achieving the desired functionality of the testing program where the calculation, display, interaction, decision, and map icons. Specific examples of how these were used follows below.

You can use the calculation icon to build custom scripting into your piece. This allows you to evaluate expressions, run functions, and keeps track of internal variables and custom variables that you create [ROBE97]. Calculation icons allow the programmer an additional level of flexibility when the basic constructs in Authorware do meet the required functionality. Calculation icons were used to overcome limitations imposed by the requirement of content modifiability. Grading and tracking of user responses is built into the Authorware system. This functionality allows for easy tracking of testing answers. This system-defined functionality and tracking ability is limited to a testing scenario that is composed of a statically defined test, i.e. the image for question one and its answer are always the same. This built-in functionality is lost due to the requirement of modifiability. The actual representation and image displayed for "`image01.jpg`" can vary from test to test. This modifiability is the desired goal of the system, and therefore statically defined correct and incorrect answers cannot be utilized within this system. As units modify what vehicles are tested and the order in which these vehicles are tested, the answers change; therefore what is a correct answer for the first image displayed on a given test generally will not be the same on successive versions taken. While this is fairly obvious, it restricts how the program can grade answers input by the user. Instead of being able to automatically set up for the same answer every time the test is run, the answers must be stored in the `userAnswer` array, and then compared to the `answerArray2` values.

If a given image in the test was always a T-62 tank, the properties response box could be set up to accept a wide range of correct answers (See Figure 17). In a dynamic environment where question one is not always the same, this ability is lost, as a properties response box must coincide with a specific question or image.


Properties: Response

T62|t62|t-62|T-62

Type: Text Entry

Pattern: T62|t62|t-62|T-62

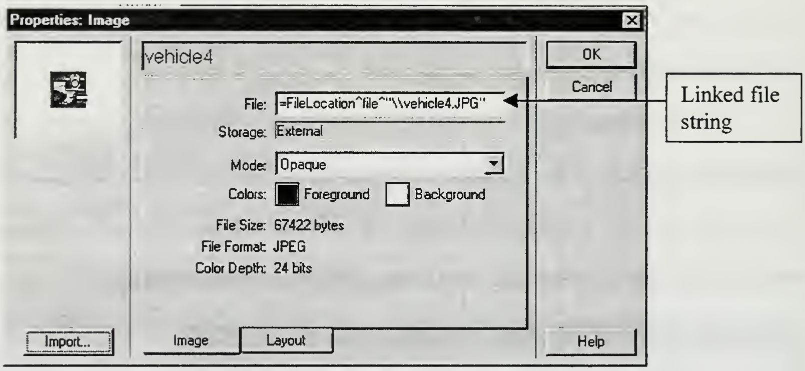
Match at Least: Words

- [ ] Incremental Matching

Ignore:
- [x] Capitalization
- [ ] Spaces
- [x] Extra Words
- [x] Extra Punctuation
- [x] Word Order

Figure 17: Response Properties Box.

Display icons are used primarily for image display purposes. As previously mentioned, Authorware allows a developer to import an image directly into the program or to link the image through a variable name. Using linked images, you can quickly update the piece by replacing the current folder where you have your images stored with a new folder of images. As long as you keep the file names of the individual linked images the same when you replace them, Authorware automatically updates your piece to reflect the current changes that you have made [ROBE97]. For the testing portion of the program the 30 display icons are linked to individual file variable names ranging from "`image01.jpg`" through "`image30.jpg`" (See Figure 18).



vehicle4

File: =FileLocation^file^"\\vehicle4.JPG"
Storage: External
Mode: Opaque
Colors: Foreground Background
File Size: 67422 bytes
File Format: JPEG
Color Depth: 24 bits

Figure 18: Image Properties Box.

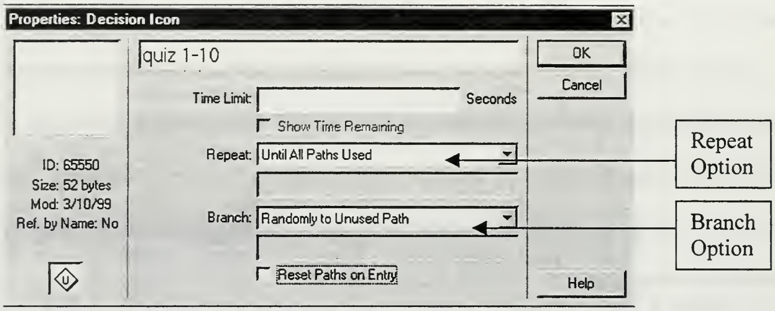
The imported file string is "`FileLocation^file^\\image04.JPG`." The variable is comprised of three variables concatenated together with the ^ operator. `FileLocation` is a system-defined variable that resolves to the directory location from which the program's executable file is being run. `File` is a user-defined variable that resolves to the directory where the photographs and answer key are located. This is the variable that is set when the user types in the test version they want to take. "`image04.JPG`" is the actual file name of the image that gets imported into the display icon. If the user runs test1 with the program from the CD-ROM drive that is assigned a drive letter D, the entire string would evaluate to "`D:\test1\image01.jpg`". To modify the program and develop a new test version, the administrator simply changes the images associated with the `image#.jpg` file names.

The interaction icon creates the majority of interactivity in your piece. The author can use it to check for user responses from a mouse, menu, or keyboard [ROBE97]. Interaction icons are used to capture the input entered by the user. In MTTS, user input is captured and assigned to the system variable `entryText`, which is then immediately transferred into the value of the `userAnswer` array element associated with that vehicle image number.

Any interaction created in Authorware is made up of four main components: the interaction icon, the response type, the response feedback, and the response branching [ROBE97]. For the testing sequence, three separate decision icons were attached to the flowline (See Figure 15). Each decision icon had ten map icons attached to it. This was done purely as a divide and conquer tool for the author, all 30 map icons could have been attached to one decision icon. Each map icon corresponds to an individual image display and answer entry sequence.

To limit the amount of time a user has to respond to each individual image, a time limit response type was added to the end of each individual map sequence. If the user has not entered a response within the allotted time, the display times out and moves on to the next image to display if there is an image that has not been displayed yet, or out of the decision icon logic if all paths have been used.

The decision icon sets up a path of options that Authorware evaluates and executes automatically on the flowline [ROBE97]. Key settings for the three decision icons containing the display logic for the test include setting the repeat and branching options of the decision icon property box (See Figure 19). These settings are the same


Properties: Decision Icon

quiz 1-10

Time Limit: Seconds

Show Time Remaining

Repeat: Until All Paths Used

Branch: Randomly to Unused Path

Reset Paths on Entry

ID: 65550
Size: 52 bytes
Mod: 3/10/99
Ref. by Name: No

Figure 19: Decision Icon Property Box.

for all three decision icons. The repeat field is set to "until all paths are used". This ensures that all ten individual images, or paths, attached to the decision icon are displayed prior to exiting the icon's looping structure and moving down the flowline. The other setting that affects the decision icon logic is the branch option setting. Options for this setting include sequentially, randomly to any path, calculated path, and randomly to unused path. Randomly to unused path allows for the randomness desired for the testing while ensuring that each question is only tested once during a given test session. The time limit field sets a time limit for the program to remain within the decision icon logic as a whole. As the program only needs to have a time limit for the individual questions within the decision icon structure, this field is not set.

The map icon simplifies and organizes the flowline by branching it into smaller segments. You can place groups of icons into a map icon to keep your code organized when creating more complex logic structures [ROBE97]. As previously stated, each of the three decision icons contain the logic to display ten images during the testing cycle. The ten icons attached to the decision icon are map icons (See Figure 15). All of these map icons contain identical logic and icons. The only difference between the individual icons is the image variable name that imports the actual digital images, and the assignment of the user's answer to the corresponding array element. The map icon contains an interaction icon, which is responsible for importing the image, and two additional map icons (See Figure 20). The first map icon attached to the interaction icon

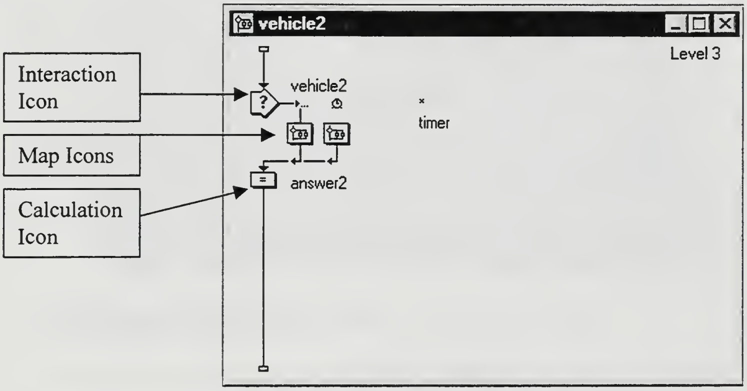
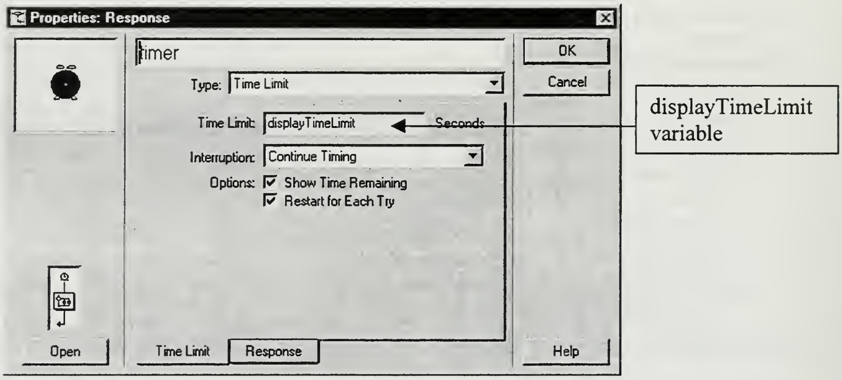


Figure 20: Structure and Icons Contained Within the Map Icons of the Decision Icon Structure.

is set for a text entry response. This is the logic that brings up the text entry box for the user to type in their answer. The * associated with this icon is a system defined wildcard. This tells the computer to accept any response typed into the entry box. The second map icon is a time limit response icon. This time limit is a "catch all" structure that limits the individual test questions to a set time limit. If no answer is typed in with the set time, this structure times out and forces the decision icon to move to the next question. The time limit allotted to this function is dynamically set by the user during the login sequence. An interaction icon catches the `EntryText` variable and assigns it to the user defined variable `displayTimeLimit`. This variable is entered in all time limit settings for the display time out values (See Figure 21). This feature allows a user greater flexibility in their training. A user who is in the early stages of their training can set it for a longer time, while advanced user can decrease the time limit allowed to view each photo.


Properties: Response

timer

Type: Time Limit

Time Limit: displayTimeLimit Seconds

Interruption: Continue Timing

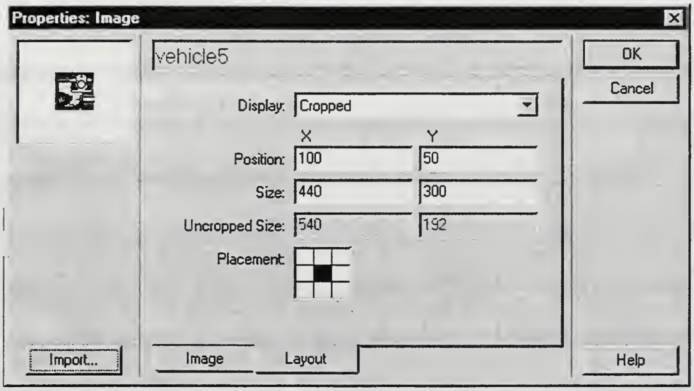
Options: ☑ Show Time Remaining

☑ Restart for Each Try

Figure 21: Property Response Dialog Box Time Limit Field.

c. Graphics and Text

The same graphics and text considerations and principles as mentioned in the trainer section were used in developing the testing program. A green background is consistent throughout the running of the program. Standard text font and size were used. All tested images are displayed in the same center screen location for all, with text entry locations for answers always placed centered below the corresponding image. The JPEG image size is limited based on the size of the overall display screen for the program. Recommended image size for use in the system is 540 by 380. This allows for an evenly divided space to both sides and above and below the displayed image. The crop feature of the image property box allows the program to accept images with a larger and smaller display size than the guideline above. If the image is too large for the given display area, the image is cropped to fit the allowable display size. The cropped portion of the image displayed is the center of the image (See Figure 22).


Properties: Image

vehicle5

Display: Cropped

	X	Y
Position:	100	50
Size:	440	300
Uncropped Size:	540	192

Placement:

Figure 22: Cropping Feature of Image Properties Dialogue Box Showing Center Placement Selection.

d. Functions and Variables

Functions and variables provide an additional level of control available to the Authorware developer. These two scripting components maintain the same general definitions as found in other programming languages. A variable stores a value that you can access any time. A function performs a certain task. These are generally contained within calculation icons, which are then placed on the flowline, in the appropriate location. There are two types of variables in Authorware: system variables, which are integrated into Authorware already; and custom variables, which you create yourself in instances where Authorware does not have a system variable defined for the data value that you want to store [ROBE97]. General function and variable concepts used are discussed below; the specific code is covered in Appendix D. Specific functions and variables relating to arrays and file I/O are covered in their respective sections.

In the MTTS, custom variables are used throughout the program for various purposes. Three examples are `userName`, `file`, and `displayTimeLimit`. The system variable `entryText` captures the entry from all text entry response types. The response in question is captured into the desired variable by using the assignment operator `:=`.

The `userName` variable is used to hold the individual user's full name. This is captured during the login sequence with an interaction icon set to accept a text entry response and a calculation icon that contains the actual code. The `userName` variable is broken into a `firstName` and `lastName` variable using the system function `getWord`.

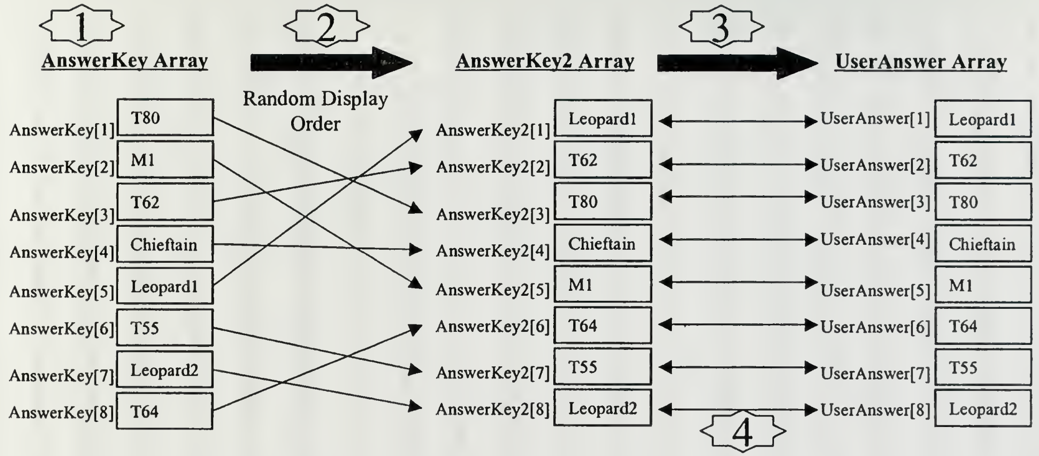
The `file` variable is used to hold the test version the user wishes to take for the given run of the program. This variable serves multiple purposes. It is the name of the directory in which the test version is stored and it also is the name of the text file that is used for the answer key.

Dynamically linking the user defined `image#.jpg` variables is the key to the modifiability of the program. As previously mentioned, these file names are linked through the variable name to their respective icons that then import them into the display screen. The program itself does not care what image is actually associated with the file name, only that it exists. When the user wants to create a different test version, two things are required. First the user must generate the appropriate answer key placed within the appropriate directory structure. Second, the user must change the `image#.jpg` files to contain the new image desired. This is covered in greater detail in Chapter IV and Appendix C.

A Repeat While system function was used to accomplish the grading of the test. Just like it sounds, this function enables you to repeat a set of instructions or series of steps a given number of times. Inside the loop, the string values of each `userAnswer` array element (the answers the user types in) are compared individually to the corresponding `answerKey2` array element. If the elements are equal, the `numberCorrect` variable is incremented. If the elements are not equal, that element number is added to the string variable of questions missed.

e. Arrays

Arrays are used to hold the individual elements of the answer key when read into the program, and to hold the individual answers to the questions when typed in by the user. Three separate arrays are initialized and used for the testing portion of the MTTS: `answerKey`, `answerKey2`, and `userAnswer`. The `answerKey` array is used to store the answer key elements read in from the answer key text file. At run time, the individual elements of the `answerKey` array are copied into the `answerKey2` array. This transfer of answers from one array into another is necessary to allow for the random order of the displayed images (See Figure 23). The `userAnswer` array holds the individual answers typed in by the user. This is accomplished by using a calculation icon in each of the 30 map icons that display the individual images. When the program enters the map icon for a given image, that corresponding image is displayed, the user types in their response, and the response is stored in the array as the array element number corresponding to the number of that map icon.



1. AnswerKey Array is filled with elements of the answerkey text file created by the administrator or user. At this point, the answers in the program and the answer key are in the same sequential order as found in the .txt file.

2. At run time, the randomness associated with the decision icon display logic picks `image05.jpg` as the first image to display. During this logic sequence, AnswerKey [5] (the answer that corresponds to `image05.jpg`) is copied into AnswerKey2[1]. In this example, the display order for the images is 5, 3, 1, 4, 2, 8, 6, 7.

3. As `image05.jpg` is displayed during run time, the user types in their answer to the displayed image. This user response is stored in UserAnswer [1].

4. This process is repeated until all images are displayed. This completes the testing sequence and the program enters the grading sequence. A string comparison between AnswerKey2[I] and UserAnswer[I] is used for grading.

Figure 23: Use of Arrays.

It is very important that the user always aligns the answer key elements with the image numbers. If the answer key contains T-62 as its first answer, then `image01.jpg` must be an image of a T-62, even though `image01.jpg` may not be the first image displayed during the actual testing sequence. As previously mentioned, the decision icon imparts a random display order for the various images attached to it. The logic of the display structure is such that whenever the system does display `image01.jpg`, the answer from the user is stored as element 1 of the `userAnswer` array. Due to random order of display, someone looking at answer key text file will only be able to determine what vehicles are going to be tested, and not the actual display order. Unit personnel will already know what vehicles they will be tested on as defined by their task, condition, and standard criteria.

f. File I/O

File I/O functions are used to both bring the answer key text file into the program for scoring and to write the user's test information into a text file for record keeping purposes. The answer key file is read into the program using the system function `ReadExtFile()`. This function uses the file variable mentioned earlier to find the correct text file to bring into the program. The entire file is stored in a variable called data. Once the answer key is in the data variable, the system functions `GetLine()` and `ArraySet()` are used to fill the individual elements of the array.

The system function `AppendExtFile()` is used to write the file information into the appropriate text file. This function has two arguments, the directory structure and file name to write the information to, and the information to write. In this case, the information to write is stored in a variable called `answerData`. This variable contains the user's answer and the answer key information for all questions presented during the test. It is written into the text file in a way that allows the user or training officer to see what particular questions were missed and need special emphasis or retraining. The printed version of this text file should be sufficient record for inspection purposes that required training was accomplished.

g. Packaging for Distribution

Packaging an Authorware piece is extremely easy. When you package a piece, the authoring menus and commands are no longer available, and the user doesn't need Authorware to run it [ROBE97]. The main option in the packaging process is whether the end product will run on a Windows platform or a MAC platform. In this case, the final product was packaged to run on the Windows platform. No distinction is made in this process between Windows 95, 98, or NT.

A clean up feature called save and compact was used prior to packaging. This save option rewrites the file to the hard disk after removing unused icon references and their associated data (stored in "byte" records). When you delete an icon from the flowline, Authorware does not automatically remove the raw data stored in the file until you save and compact it [ROBE97]. Depending on how many modifications are made during the production of the piece, this can help reduce the executable file size. The executable file size for this thesis was 1.4 MB. The author does not feel that this file size will increase greatly with future modifications of the system. This is due to the fact that the image files are external to the actual program. 30 JPG image files are shipped with the product in a tutorial folder. These images are used when setting up the first test (See

Appendix C). An additional 200 digital images were shipped with the product to use as replacements for test development at the unit level.

A finished product typically consists of not only the Authorware file that contains the flowline but also external files, Xtras, scripting Xtras, and DLLs that the Authorware file may use. To run properly, a packaged piece requires that these external files be available. If they aren't already available on the computer running the packaged piece, you must provide them [MACR98]. Four Xtra files must be included with the distribution of the executable file for the program to run on the user's computer. These were all related to image reading for the program. The Xtras `Mix32.x32` and `Viewsvc.x32` are required for any type of image format. The Xtras `Jpegimp.x32` and `Mixview.x32` are specific Xtras required for the JPEG format. The developer must create a separate Xtra folder in the same directory as the program executable file. All Xtras required for the program to work correctly must be placed in this folder and shipped with the final product.

3. Limitations

The question of how much time to allow a user to type in their answer is an issue that must be addressed. The ability to correctly identify weapons systems in both training and real world environments is a timed task; however, the keyboard entry ability of individuals varies greatly. It was initially thought that this could be overcome by having the images displayed for the standard amount of time, and have the text entry box stay active for additional amount of time. At the time of writing, the author could not make this functionality work. This seems to be a limitation in the flowline structure of the authoring tool. The current MTTS allows the user to vary the amount of time an image is displayed during the testing sequence. This amount of time is also the amount of time that the user has to type in his or her answer to the displayed image. These times run concurrently. Allowing the user to modify this time will help overcome the differing abilities of keyboard proficiency of the individual users. An adjustable time setting also allows for the natural progression of skill in the given test subject. Individuals who are learning and testing new material can set this time limit to a high value. This will allow them extra time to study the material, while still early in the learning process. Once the user has developed a given level of proficiency on the study material, this time limit can be set to a more challenging level. In the case of unit threat identification training, this time limit will generally be a standardized value based on the requirements given in individual unit SOPs. This issue is addressed further in Chapter VI.

F. SUMMARY

This chapter covered the application of the ISD process and how it was used to help develop the MTTS. Key features of Macromedia Director 6.0 and Authorware Attain 5.0 that were used in developing both the training and testing portions of the MTTS were covered. Limitations of both software systems discovered by the author in the course of his research were also addressed. Specifics on using the MTTS and how to tailor it to specific unit needs are topics covered in the following chapter.

IV. USABILITY

A. INTRODUCTION

This chapter covers issues of usability for both the training and testing portions of the MTTS. It specifically covers how to install and run both programs, program output, and how individual units can tailor the testing portion to meet their specific training requirements. It then discusses issues of program upgrades and maintenance.

B. USING THE PROGRAMS

1. Distribution

For the MTTS to run on a user's computer, two main executable files must be copied onto the user's hard drive: the Director movie executable file (MTTS training program), and the Authorware Attain 5.0 executable file (MTTS test program). The desired images to use for the Authorware test program and a folder containing the required Xtra files must also be copied into this same hard drive location. In its current form, the sizes of the Director and Authorware executable files are 70 MB and 1.5 MB, respectively. The total size of the digital images used in the test version shipped with the tutorial package is 25 MB. The only file whose size will remain fairly constant is the Authorware executable. The size of the Authorware executable file does not include the image files for the 30 images used during the testing sequence (these images are brought into the program at run time). The total amount of images shipped with the product can vary, and will more than likely increase. This will give the unit a greater ease and flexibility in developing a variety of tests covering their respective threat scenarios. In its current state, the training portion has one of seven categories completed (Armor). The logic structure for the remaining six categories is in place; only the images and text information need to be incorporated into the system. Once the training portion is completed and contains full information for all categories of vehicles and equipment, the executable file size will increase and require the use of a CD-ROM for distribution. It is estimated that the complete MTTS size will be roughly 280 MB.

2. Running the Programs

Clicking on their respective .exe file from the Windows explorer menu runs both programs. The main factors on where the executable files are run from are available space and speed of the peripherals (e.g., CD-ROM drive) on the user's system. The optimal method is for the user to load the files onto their hard drive, as the programs will incur the least amount of delay when run in this mode. If space is a limiting factor, the programs may be run directly from the distribution medium. In this case, the transfer rate of the peripherals on the given system will affect how fast the programs run. In general, when the testing program was run from a 100 MB ZIP disk, there was roughly a one-second delay for the images to appear on the display window. This is in comparison to no noticeable delay when run directly from that computer's hard drive.

3. Program Output

The training portion of the MTTS has no output. The testing portion creates an output text file for each run conducted. This text file is named with the user's last name and stored in the same directory as the testing executable file. Information concerning individual test runs is stored in this file. If a text file for the test user does not exist (either it is the user's first time running the system, or the text file for that user has been deleted), the appropriate text file is created. When a user takes successive test versions, their original output text file is appended with the new information. The purpose of this text file is two-fold: the file can be printed and stored in the individual's training record as a indication that required training has been accomplished, and it can provide necessary focus on what vehicles and equipment need emphasis during training sessions. In its current form, the MTTS does not provide any counter measures to ensure that individual users do not tamper with their user text files, i.e. change answers to increase their score. In a "for record testing event" the training officer responsible for administering the training can monitor the test and immediately view scores. See Figure 24 for an example of testing file output.

```
*****************************************************
CPT Doug Miller took test version: pretest
Test date and time: 7/26/99 8:24 PM
Start time was 8:23 PM. Total time on the system this run was 0:01
Total number of correct answers on this run: 30
Total number of incorrect answers on this run: 0
Questions missed were: None
Question #    Answer Key    User Answers
  1           leopard2      leopard2
  2           t80           t80
  3           challenger    challenger
  4           m1            m1
  5           t55           t55
  6           leopard1      leopard1
  7           chieftain     chieftain
  8           m1            m1
  9           t62           t62
  10          challenger    challenger
  11          leopard1      leopard1
  12          t72           t72
  13          leopard2      leopard2
  14          t64           t64
  15          t80           t80
  16          t55           t55
  17          t64           t64
  18          t72           t72
  19          t62           t62
  20          t80           t80
  21          chieftain     chieftain
  22          leopard2      leopard2
  23          m1            m1
  24          t64           t64
  25          challenger    challenger
  26          t55           t55
  27          leopard1      leopard1
  28          t72           t72
  29          t62           t62
  30          chieftain     chieftain
*****************************End of Run*************************
```

Figure 24: File Output from Sample Test Run.

C. TAILORING CONTENT

1. External Linking of JPEG Images

A main goal of this thesis was to develop a system that individual units and users could modify to their own specific training needs. This was accomplished through the use of the Authorware software feature of dynamic linking of images to the individual display icon logic. The digital images used during the testing portion of the system are dynamically linked through the use of variable names to the display icons at run time. For the system to recognize the images, they must be of the JPEG format standard.[1] To convey maximum content variability, the variable naming convention for these images is simply "`image#".jpg`, i.e. `image01.jpg`, `image02.jpg`, ...`image30.jpg`. In this manner, the tested content can be images of vehicles, planes, ships, or virtually any type of image the user wants to test. As long as the user has also developed an answer key for grading the individual responses to the displayed images, the program will work. For system development, the author concentrated his effort on ten different armor vehicles. Three digital images of each vehicle type were used. One of the armor vehicles used was the M1 tank. For use in the program, the three different images of the M1 used were labeled `image04.jpg`, `image08.jpg`, and `image23.jpg`. In the answer key that goes with this test version, the fourth, eighth and twenty-third answers were all simply m1.

The image file size for images used in program development ranged from 30 KB to 70 KB. There is no limit to the file size of the images used in this program; however, the display size of the image is limited to 440 x 300 pixels. The resolution of the

[1] This is the standard image format adopted by the author; Authorware can read a wide variety of image formats.

displayed image during a test run corresponds directly with the resolution quality of the image imported into the program. A digital image that has high resolution will look better when displayed than an image that has a lower resolution.

2. Importing the Answer Key

For the testing program to grade the user's answers, an answer key .txt file must be made and stored within the same directory file structure as the executable file. This file can be developed in any of the numerous text editor or word processing programs available. Answers must be typed in on separate lines of the file, separated by a return key entry. This name of this file must be the same as the name of the test the user types in when the program asks the user what test they want to take. This file is brought into the program with each answer becoming a separate element of the AnswerKey array. It is important to note that due to the previously mentioned process of program grading that users are made aware of how answers are entered into the answer key. Specifically whether answers are lower case, upper case, include hyphens etc. A string comparison is made between the individual answer key array and the user answer array elements. If a user enters their responses in all upper case and the answer key is generated with all lower case, even if the user identifies the vehicle correctly, it will be scored incorrectly. A quick glance of the output text file will determine whether or not an incorrect answer was the result of this type of error. The answer key used to develop this system is all lower case and does not include hyphens, i.e. m1, chieftain, t80 (See Figure 25).

```
leopard2
t80
challenger
m1
t55
leopard1
chieftain
m1
t62
challenger
leopard1
t72
leopard2
t64
t80
t55
t64
t72
t62
t80
chieftain
leopard2
m1
t64
challenger
t55
leopard1
t72
t62
chieftain
```

Figure 25: Example Answer Key .TXT File Imported into Testing Portion of System.

D. PROGRAM UPGRADES AND MAINTENANCE

1. Upgrades

One of the advantages that a CBT system offers over the traditional FM 1-402 approach is the ability to quickly modify course content and distribute these changes in a timely fashion. As has been demonstrated by the civilian software market, upgrades to software packages can be marketed as often as the material warrants. With the threat identification CBT, the basic logic structure is in place. If new or varied coverage of a particular vehicle or weapon system is desired, that information must simply be placed into this pre-existing structure, tested for correctness, and then a new executable file can be packaged and distributed

2. Maintenance

Very little in the way of file maintenance is required of the user. Other than the basic files described in the tutorial in Appendix C, no other files are required to reside on the users system. Currently there is no automatic installation feature; all files are manually put on the hard drive by the user through the use of copy and paste commands in the explorer window. When the user wants to remove the program from their computer, all they have to do is delete the appropriate folder. There is no uninstall feature in the MTTS current form.

When numerous personnel are going to use the test program on a given computer, the test program will generate an individual text file for each user. These text files contain the grading information for the individual user. As a user continues to run through more tests, this file is appended with the information from each additional test run. These files can be printed for record and then deleted if desired.

E. SUMMARY

This chapter covered usability aspects of the MTTS. General usage instructions, tailoring content, maintenance and upgrades, and feedback from test users were all discussed. Specific instructions on setting up the MTTS are included in Appendix C. Comparison of the MTTS versus FM 1-402 is covered in the following chapter.

V. STATISTICAL TEST OF THE MTTS TRAINING SYSTEM VERSUS FM 1-402

A. INTRODUCTION

This chapter covers the experiment conducted to compare the quality of training of the MTTS training system versus FM 1-402. The purpose of the study was to compare the MTTS training system described in chapter III to FM 1-402 as methods of training soldiers on individual threat identification tasks.

B. PURPOSE AND RATIONAL

The hypothesis of this experiment is that the MTTS training system is at least as good as FM 1-402 for training soldiers on individual threat identification tasks. In this experiment the use of the MTTS system was measured against that of an existing Army method of training threat identification. This experiment manipulated the training medium that the participants use during a fixed length study period. The control group studied Field Manual 1-402, also called The Aviator's Recognition Handbook. The test group studied using the MTTS training system. Both groups were allocated the same amount of study time (20 minutes) prior to taking the threat identification test at the end of the study period. Due to the interactive nature of CBT, it was expected that the test group would produce higher post-test scores than the control group, thus demonstrating that a CBT application is a suitable substitute for current Army methods of training individual soldiers on threat identification training tasks.

C. METHODS

1. Participants

Fifty-seven officers, ranging in age from 29 to 39 years, served as participants for this experiment. The group consisted of Army, Navy, and Marine Officers currently attending the Naval Postgraduate School or the Defense Language Institute. Countries represented in the study were the United States, Germany, Turkey, and Australia. The amount of threat identification training participants had prior to the experiment varied widely. The Naval officer participants had little to no prior training on armor vehicle recognition. Some Army officer participants were Armor or Aviation Officers with extensive training on armor threat identification. Some participants had used FM 1-402 to train threat identification tasks in previous military assignments. No participants had utilized the MTTS prior to the experiment.

The participants were all competent with computers and navigation using graphical interfaces similar to that of the MTTS system used in the study. A majority of the participants were computer science majors with experience in numerous programming languages and graphical user interface applications. The participants that were not computer science majors were in areas of study that required frequent use of computers in their given fields of study. Participants were assigned to the test or control group based on the last digit of their social security number (See Appendix A for test and control group participant data).

The independent variable for the research was manipulation of the training medium used by the participants during the 20-minute study period. The control group studied FM 1-402 while the test group studied using the MTTS training system.

2. Materials and Apparatus

The experiment consisted of three phases. Phase one consisted of a threat identification pre-test. Phase two was a study period of 20 minutes. Phase three concluded the experiment with a final threat identification test.

The phase one pre-test material was a Microsoft PowerPoint presentation consisting of thirty black and white photographs of friendly and threat tanks. Each photograph occupied one PowerPoint slide (See Figure 26). Three separate photographs of ten different armor vehicles were used for the presentation. The slides were displayed in slide-show format with each slide on screen for fifteen seconds. The slides of the test were presented to the participant in a random order. This order was determined prior to the first experiment run and remained the same for all participants.

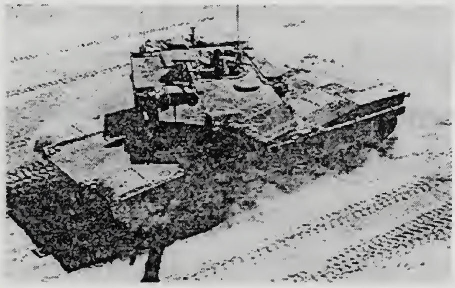


Figure 26: PowerPoint Test Slide Example.

Phase two study materials differed for the control group and test group. The control group studied Army Field Manual 1-402. This manual contains three black and white photographs and a written description of each vehicle (See Figure 1). The test group utilized the MTTS training system. The MTTS allowed participants to view multiple photos of each vehicle, and read the written description from the Field Manual. Each vehicle displayed in the MTTS had three to five digital photographs that the participant could view. Each photograph had various recognition feature statements highlighting various features visible in that photograph.

Phase three final testing used the same PowerPoint slide show format as in phase one of the experiment to test friend or foe and nomenclature information. As with the pre-test, the presentation order of the vehicles was randomly determined prior to the experiment and remained constant for all participants. Neither group had seen the photographs of the armor vehicles in the phase three test in any previous phase of the experiment.

A Dell Dimension XPSR 400 computer with a ViewSonic model P815 21-inch color monitor was used for phase one and three testing, and the test group phase two study period. The monitor resolution was set at 1024 by 768 pixels with a 16-bit high color setting.

3. Procedure

To begin the experiment, participants were briefed on the requirements of the study and the particular phases of the experiment. After the briefing, the experimenter asked the participants if the last digit of their social security number was odd or even. Participants with an odd numbered last digit were placed in the control group; participants with an even numbered last digit were placed in the test group. During the course of the experiment, if one group started to get too large relative to the other group, participants were placed into the group with fewer numbers until the groups became even in numbers tested.

Once the participant's group was determined, the experimenter administered the phase-one threat identification test. During the test, the experimenter observed the presentation to ensure that all slides were present on the screen for the stated 15-second interval. During this time, the student was required to identify the friend or foe status based on a Warsaw Pact versus NATO threat model, and to indicate the nomenclature or name of the vehicle. Participants wrote their answers on a provided answer sheet (See Appendix B). This phase of the experiment was the same for both the control and test group participants.

During the phase two study period, the control group participants were given twenty minutes to study FM 1-402. The experimenter explained the layout of the manual, indicated what vehicles were going to be tested in phase three, and answered any questions the participants had. The test group participants were given twenty minutes to study using the MTTS training system. The experimenter explained the layout of the application, how to navigate to appropriate pages of the application, what vehicles were going to be tested in phase three, and answered any questions the participants had. Once the twenty-minute clock started however, the experimenter provided no assistance to the participants of either group. At the completion of the twenty minutes, the experimenter informed the participants that the study period was over and that phase three testing would begin.

The experimenter then administered the phase three threat identification test to the participants. The experimenter reiterated what information was going to be tested, the format of the test, and the format of the answer sheet. The experimenter then started the test and again observed the presentation to ensure the slides were displayed for the correct amount of time. At the completion of the phase three test, participants were thanked for their participation and debriefed as to the purpose of the experiment.

D. RESULTS

When a participant finished the experiment, the difference between their post-test and pre-test scores for both friend and foe identification, and nomenclature was determined. The summary statistics for both control and test groups are listed in Table 2.

	Friend or Foe Identification		Nomenclature	
	Mean score increase	Variance of increase	Mean score increase	Variance of increase
Test Group	2.92	11.07	9.33	17.31
Control Group	1.44	9.26	6.74	21.12

Table 1: Experiment summary statistics. The test group had a greater mean increase in both categories than the control group.

For statistical analysis, a two-sample t test using a pooled estimator of the variance was performed [DEVO95]. The difference between the averaged post (μ_1) and pretest (μ_2) scores was compared separately for the friend or foe identification and nomenclature data. The null hypothesis for the experiment was that $\mu_1 - \mu_2 = 0$ for both the nomenclature and friend or foe data sets. For the friend or foe identification data, differences between test and control group post-test scores failed to achieve statistical reliability, $t(50) = 1.68$, $p > 0.05$.[1] Forty-eight participants scored within five points of the maximum possible score on the final friend or foe portion of the test. Of these, twelve scored the maximum allowed. It is therefore likely that a ceiling effect reduced the statistical power for the friend or foe data. For the nomenclature data, the null hypothesis was rejected in favor of the alternate hypothesis; the average post-test scores were higher

[1] Due to data corruption 2 test scores were deleted from the control group friend or foe data set.

for participants who had studied using CBT than for those who had studied FM 1-402, $t(52) = 2.17$, $p = 0.03$.

Since the goal of the experiment was to determine the suitability of using the MTTS as a training medium for threat identification training, failing to reject the null hypothesis for the friend or foe identification data is not seen as a negative result. While the test was not statistically significant, the average difference between pre-test and post-test scores of the test group was noticeably higher than the control group. At the very least, it appears that the use of the MTTS is at least as good as the field manual in this training task. The result of the nomenclature test is stronger evidence in favor of using the MTTS for threat identification training purposes.

E. SUMMARY

This research demonstrated that CBT is a viable method for threat identification training. The identification of friend or foe testing did not prove a statistically reliable increase in benefit of MTTS over FM 1-402; however, the average increase was higher for participants in the test group. Test group participants did show statistical significance over the control group participants in the nomenclature testing. Taken together, results demonstrate that the MTTS is a viable medium for threat identification training.

The MTTS used in the experiment had limited multimedia content in its presentation. There were no video, audio or animation sequences to instruct the participant. Hailey and Hailey found that "looking only at aesthetics and ease of use, digital multimedia may be preferable to traditional multimedia" [HAIL98]. Based on the results of this study, it would appear that even a basic CBT may be preferable and increase amount learned over a written training medium.

However, increasing the multimedia content of the MTTS to more fully engage the individual in the learning process might increase the amount learned and the efficiency of the system. Possible advances to this system include: using audio to identify recognition features, increased photograph quality, charts of key recognition feature differences, and the ability to compare a photograph of one vehicle with different vehicles to highlight differences in subtle recognition features. It appears that by using the MTTS system to train threat identification, units could increase soldiers' proficiency in this task and help reduce the number of fratricide events in future conflicts.

VI. SUMMARY AND RECOMMENDATION FOR FUTURE RESEARCH

A. INTRODUCTION

The goal of this thesis was to develop a PC-based, multimedia-oriented, modifiable threat identification system for individual training and testing tasks. The system must be easy to use with content that is easily modifiable at the unit level. Training with the system should produce at least the same amount of learning and retention as training with FM 1-402, a currently used Army threat identification training manual. Using Macromedia Director 6.0 and Authorware Attain 5.0 software the goals were achieved.

B. SUMMARY OF WORK

Background research was conducted to develop a background of CBT, military training, fratricide, and current requirements and methods of threat identification training in Army Aviation units. The five-phase ISD process was studied and then applied to develop the MTTS. The ISD process and how it was applied to this work were addressed in Chapter III. The MTTS is divided into two subsystems: training, and testing.

The MTTS training subsystem was developed using Macromedia Director 6.0 software. This software allows for the incorporation of a wide variety of multimedia formats into the developed system. For this project, multimedia content was limited to digital images, sound files, and text. The training subsystem was used in the statistical comparison of the MTTS versus FM 1-402 in training threat identification tasks. The MTTS training subsystem and the specific Macromedia Director 6.0 functionality used in developing it are covered in Chapter III, section D.

The MTTS testing subsystem was developed using Macromedia Authorware Attain 5.0. This software is a more dedicated computer-based-testing development application. Greater modifiability was achieved using this software as it allows for easier incorporation of linked media through the use of user-defined and system-defined variables. The MTTS training subsystem is covered in more detail in Chapter III, section E.

Initially there seemed to be a fairly large division between the training and testing subsystems of the MTTS and their functionality. During the course of the work, however, it was discovered that the testing subsystem could provide benefit in the training aspect of threat identification.

A statistical test of the MTTS training subsystem versus FM 1-402 was conducted to determine whether using CBT in general is a viable training method for threat identification training. It was shown that using the MTTS provides at least the same amount of training quality as FM 1-402. This was covered in detail in Chapter V.

C. RESEARCH QUESTIONS

The following questions were addresses in this thesis:

- *What are the current methods of training threat identification tasks in Army Aviation Units?*

  Currently Army Aviation units use FM 1-402 as a means of training threat identification tasks. This manual was published in 1984 and is not modifiable at the unit level. The CBT system ROC-V is available, but is primarily geared for use by ground personnel, and is not modifiable at the unit level.

- *Can a multimedia threat identification training system enhance this training and thereby increase the performance of soldiers when required to correctly identify threat vehicles in a hostile situation?*

  The statistical test comparing the MTTS training subsystem to FM 1-402 shows that CBT is at least as good as FM –1042 in training threat identification tasks.

- *How can Macromedia Director 6 software increase the quality of threat identification training for Army Aviation Units?*

  Macromedia Director 6.0 software was used to develop the MTTS training subsystem. This software allows for the development of CBT executable files that can run on individual PCs. These CBT products can contain a wide variety of multimedia content. It is important to remember that to have the greatest training benefit, the multimedia content must be linked with a quality training strategy.

- *How can the Macromedia Authorware 5.0 Attain software increase the quality of threat identification training for Army Aviation Units?*

  Macromedia Authorware Attain 5.0 software was used to develop the MTTS testing subsystem. This software package is a more dedicated testing development suite. Success was achieved in developing a system that is modifiable at the unit level. Units can modify the images and text file answer key that the executable file imports at run time, thus allowing great flexibility in what threat vehicles individuals in the unit are tested on. Modifiability will greatly aid in meeting threat identification training requirements in a deployment situation.

D. RECOMMENDATIONS FOR FUTURE WORK

The following are areas of research that the author feels would be beneficial to furthering the research efforts of this thesis.

1. Increased Multimedia Content

Further research concerning increased multimedia content in the MTTS would be beneficial. Specific research questions could include:

- Does the use of video and increased sound information for recognition features increase or decrease the learning and retention processes?
- How much multimedia content is enough?

2. Increased Modifiability in MTTS Training Subsystem

The training subsystem did not achieve the same level of modifiability as the testing subsystem. Specific research questions could include:

- Can Macromedia Director 6.0 support the level of modifiability desired?
- Can Macromedia Authorware Attain 5.0 be used to develop the training subsystem of the MTTS?

3. Human Computer Interaction (HCI)

Numerous HCI issues can be addresses in further research. For example:

- Can voice recognition software be integrated into the testing subsystem of the MTTS?
- How does navigation and content presentation affect retention in CBT use?

4. Virtual Environments for Threat Identification Training

It is hoped that future work would include the use of virtual environments in the training of threat identification tasks. Along these lines, future studies should address positive or negative habit transfer of existing virtual environments that are used in currently existing virtual environment training systems such as SIMNET.

APPENDIX A. TEST AND CONTROL GROUP DATA

A. TEST GROUP DATA

Subject #	Service	Pre-test Score F/F	Final-test Score F/F	Difference F/F	Pre-test Score NOM	Final-test Score NOM	Difference NOM
2	USMC	20	28	8	0	15	15
3	USA	30	29	-1	18	26	8
4	USMC	24	28	4	8	13	5
5	USN	14	26	12	0	12	12
8	USA	28	30	2	14	22	8
9	USMC	28	29	1	9	16	7
10	USA	26	30	4	11	18	7
14	USA	27	29	2	10	20	10
16	USA	30	30	0	18	23	5
18	Turkish Navy	16	21	5	0	16	16
21	USA	27	30	3	8	15	7
23	USMC	20	28	8	1	12	11
25	USMC	23	27	4	3	18	15
26	USA	29	29	0	11	22	11
28	USA	23	26	3	6	19	13
29	USA	16	24	8	3	16	13
32	USMC	30	30	0	24	25	1
34	USA	30	28	-2	13	17	4
36	USN	24	25	1	0	6	6
38	German Army	30	30	0	20	26	6
40	USMC	27	28	1	14	22	8
41	USN	21	26	5	0	16	16
43	USA	28	29	1	2	10	8
48	USMC	22	28	6	0	16	16
49	USA	27	29	2	11	16	5
50	USA	27	26	-1	12	19	7
52	USA	26	29	3	3	15	12

B. CONTROL GROUP DATA

Subject #	Service	Pre-test Score F/F	Final test Score	Difference F/F	Pre-test Score NOM	Final-test Score NOM	Difference NOM
1	USMC	22	25	3	1	8	7
6	USA	30	30	0	23	28	5
7	USN	26	29	3	4	12	8
11	USMC	16	23	7	1	9	8
12	USMC	27	29	2	4	16	12
13	USA	28	28	0	21	19	-2
15	USMC	19	29	10	1	15	14
17	USA	30	30	0	19	24	5
19	USN	28	28	0	1	11	10
20	USA	28	29	1	15	18	3
22	USMC	23	24	1	0	13	13
24	USN	28	30	2	3	14	11
27	USA	26	27	1	7	15	8
31	USMC	29	30	1	22	22	0
33	hilean Na	22	28	6	6	11	5
35	USA	30	30	0	17	19	2
37	USN	28	27	-1	8	9	1
39	USN	22	23	1	0	6	6
42	wegian N	25	27	2	2	15	13
44	USN	1	28	*	0	7	7
45	tralian Ar	5	27	*	0	11	11
46	USMC	27	23	-4	8	9	1
47	USMC	27	29	2	8	15	7
51	USA	30	30	0	20	21	1
53	USA	30	26	-4	9	12	3
54	USMC	30	29	-1	7	21	14

* The data for these participants was corrupt and discarded for the friend/foe statistical analysis portion of the experiement.

APPENDIX B. PRE- AND POST-TEST ANSWER SHEET EXAMPLE

Threat Identification Pre-Test

****

Subject Information

Subject Number: ________________

Name: ________________________

e-mail: _______________________

Date: ________________________

Branch of Service: _______________

Years of Service: ________________

****

Instructions

You will be shown 30 different photographs of friendly and enemy armor vehicles. You will see some vehicles more than once during the presentation. For each photograph shown please answer the following: Whether the vehicle is considered friend or foe in accordance with a soviet-based model, and the nomenclature of the vehicle. See below for example.

Vehicle Number :

a. Friend or Foe
b. Nomenclature: ________________________________

At the end of the pre-test, you will be allowed to study for twenty minutes.

Once the twenty-minute study period is finished, you will be given another test. During this test, you will be asked to provide the same information as in the pre-test.

Vehicle Number 1:

a. Friend or Foe
b. Nomenclature: ______________________________

Vehicle Number 2:

a. Friend or Foe
b. Nomenclature: ______________________________

Vehicle Number 3:

a. Friend or Foe
b. Nomenclature: ______________________________

Vehicle Number 4:

a. Friend or Foe
b. Nomenclature: ______________________________

Vehicle Number 5:

a. Friend or Foe
b. Nomenclature: ______________________________

Vehicle Number 6:

a. Friend or Foe
b. Nomenclature: ______________________________

Vehicle Number 7:

a. Friend or Foe
b. Nomenclature: ______________________________

Vehicle Number 8:

a. Friend or Foe
b. Nomenclature: ______________________________

Vehicle Number 9:

a. Friend or Foe
b. Nomenclature: ______________________________

APPENDIX C. TUTORIAL

A. INTRODUCTION

This appendix covers a brief tutorial on how to run both the training and testing portions of the developed CBT. The steps for running and using the training portion are fairly straightforward and covered step-by-step. A fictitious scenario was developed to guide the user through the development of a unit specific testing program. After running through the steps in this tutorial, the user should be able to apply the concepts learned to develop their own tests specific to their given situation. The last section of this appendix contains a checklist and a troubleshooting guide to aid the user when developing additional test versions.

B. TRAINING SYSTEM TUTORIAL

This section covers instructions on how to install and run the MTTS training subsystem. General navigation functionality is addressed as well.

1. Installation and Running

The MTTS training subsystem can be run either directly from the CD-ROM, or it can be installed to a local hard drive and run from there. Due to the lower data transfer rate of peripheral devices, it is recommended that the program be run from a local hard drive. In either case, double click on the `MTTSTrainer.exe` file to start the program. If the program is to be run from a local hard drive, use the copy and paste commands to transfer the `MTTSTrainer.exe` file to the desired location.

2. Navigation

Navigation in the MTTS training subsystem is accomplished by using the mouse to click on appropriate locations on the viewing screen. See Figures 27 through 29 for navigation features.

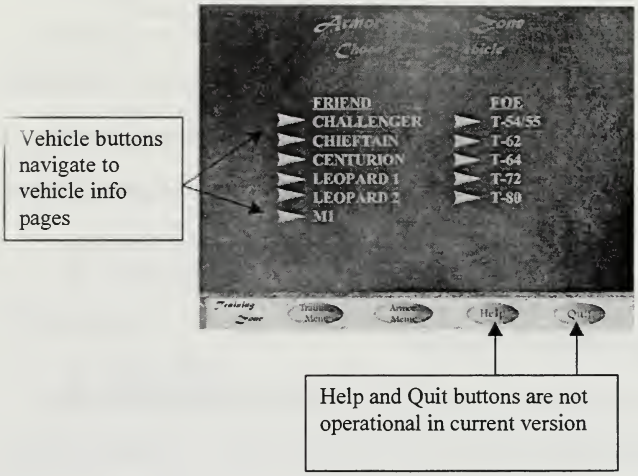


Figure 27: MTTS Trainer Armor Main Page.

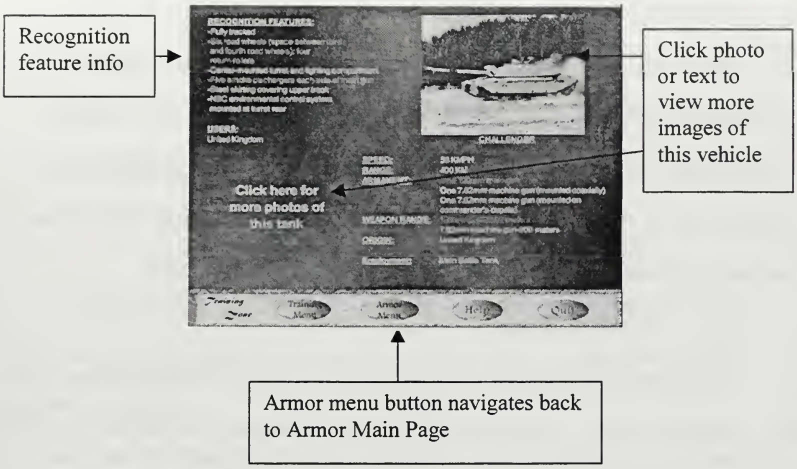

Recognition feature info
Click photo or text to view more images of this vehicle
Click here for more photos of this tank
CHALLENGER
Training Zone
Training Menu
Armor Menu
Help
Quit
Armor menu button navigates back to Armor Main Page

Figure 28: MTTS Trainer Vehicle Main Page.

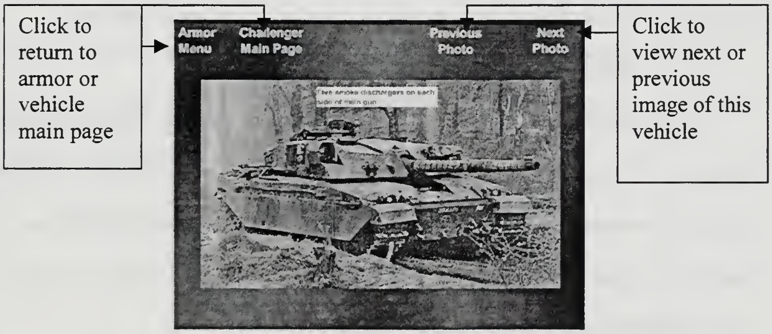

Click to return to armor or vehicle main page
Armor Menu
Challenger Main Page
Previous Photo
Next Photo
Five smoke dischargers on each side of main gun
Click to view next or previous image of this vehicle

Figure 29: MTTS Trainer Vehicle Sub-Pages.

C. TESTING SYSTEM TUTORIAL

This section covers step by step procedures to follow for developing a test version and running the system as a user. It assumes a basic understanding of the Windows 95/98/NT Operating System to include: copying and pasting files, renaming files, and navigation through the explorer window. It should take approximately 20-25 minutes to complete the tutorial and take the test version created during the tutorial.

1. Test Development

This tutorial develops a fictitious deployment scenario for a non-existing Army unit. A brief background section covers basic enemy information for the user. Following this, a step by step guide will take the user through the required procedures to develop a test version that could be used to test deploying personnel on the specific threat in the deployment region. Once the user completes the tutorial provided, he or she will have the necessary tools to develop tests specific to their individual unit's missions.

2. Scenario Development and Tutorial Steps

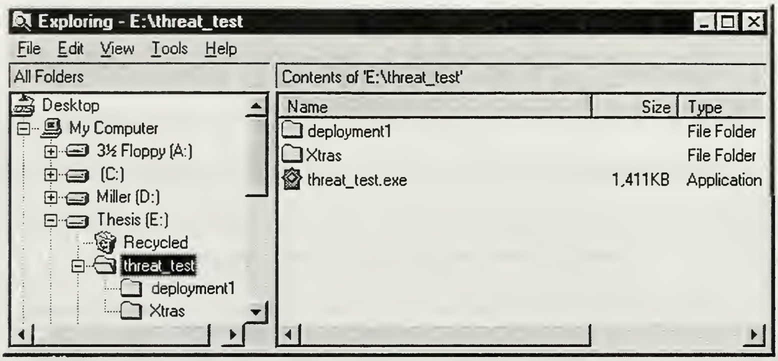
You are currently assigned to an Attack Helicopter Battalion that has been alerted to deploy to the country of (NTC threat country). As part of your unit's pre-deployment training, all crewmembers are required to complete threat identification training on the threat they will face once deployed. The S-2 has disseminated the friendly and threat order of battle, which is consolidated in Table 4.

WEAPON SYSTEM	ENEMY	FRIENDLY
ARMOR	T-80, T-72, T-64	M1, CHIEFTAIN
APC	BRDM-2, BTR-60, BTR-70	M2/M3, LAV-25
ROTARY WING	MI-4, MI-8, MI-24	AH-64, UH-60

Table 2: Enemy and Friendly Forces Composition.

Based on the above information, you decide to make a threat test that contains two photos for each of the above weapon systems. Follow the instructions below to complete this project.

1. Ensure that the Hide MS-DOS file extensions for file types that are registered box is unchecked. To verify this, navigate to the explorer window, click on view, click on options, and then click on the view tab.
2. Create a folder named `threat_test` on your computer's hard drive.
3. Use the copy and paste command to copy the following files from the `Tutorial` folder on the CD-ROM that came with this system, into the `threat_test` folder that you created in the step two:
   - `threat_test.exe` (an individual file)
   - `Xtras` (a folder that contains 4 .xtra files)
4. In the `threat_test` folder created in step two, create a sub folder named `deployment1`. At this point you should have a directory structure in place that matches that of Figure 30.



Contents of 'E:\threat_test'

Name	Size	Type
deployment1		File Folder
Xtras		File Folder
threat_test.exe	1,411KB	Application

Figure 30: Example of Directory Structure and Files on User's Hard Drive.

5. Use the copy and paste command to copy the file `Answerkey.txt` (in the `Tutorial` folder) into the `deployment1` folder created in step four. This file is currently zero in size. You will add the required information to this file in step eight.

6. Determine the order for the images you are testing, in this case, the 30 jpeg images of vehicles. This is not necessarily the display order of the images when the test is run, this step establishes the relationship between the answer key text file, and the images displayed in the program. In this example the order is as follows: T-80, M1, BRDM-2, AH-64, MI-4, T-72, MI-8, BTR-70, M2/3, LAV-25, T-64, M-113, CHIEFTAIN, UH-60, MI-24, CHIEFTAIN, T-80, LAV-25, BRDM-2, MI-4, UH-60, T-72, M1, MI-8, M2/3, BTR-70, M-113, AH-64, T-64, MI-24

7. Go to the folder named `Tutorial_Images` (in the `Tutorial` folder). This folder contains two digital JPEG images of all weapons systems for the test you are developing.

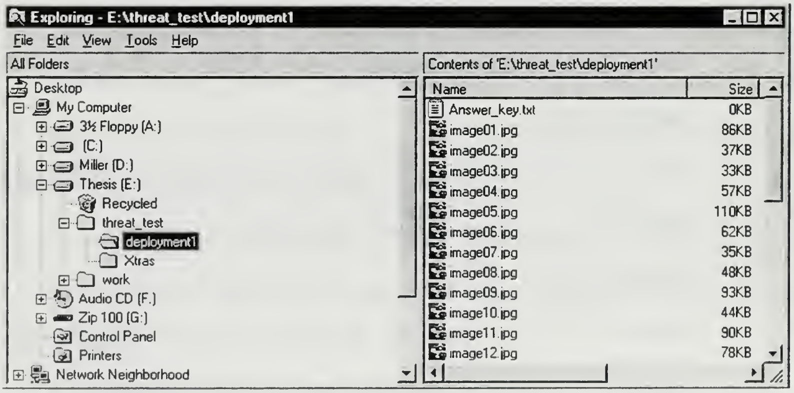
    - Copy the thirty separate image files from the `Tutorial_Images` folder into your `deployment1` folder

8. To help the renaming process in this step, it is recommended that you sort the list of file names in the explorer window by name. (With the deployment1 folder highlighted, use the view, arrange icons, by name commands in explorer). Rename the .jpg files according to the test order developed in step six above. This renaming creates a link between the images and the corresponding answer in the answer key file created in the step nine. (Renamed files are in bold-faced text for ease of viewing during the renaming process).

    - Rename `ah64_1.jpg` to **`image04.jpg`** and `ah64_2.jpg` to **`image28.jpg`**
    - Rename `brdm2_1.jpg` to **`image03.jpg`** and `brdm2_2.jpg` to **`image19.jpg`**
    - Rename `btr70_1.jpg` to **`image08.jpg`** and `btr70_2.jpg` to **`image26.jpg`**
    - Rename `chieftain_1.jpg` to **`image13.jpg`** and `chieftain_2.jpg` to **`image16.jpg`**
    - Rename `lav25_1.jpg` to **`image10.jpg`** and `lav25_2.jpg` to **`image18.jpg`**
    - Rename `m1_1.jpg` to **`image02.jpg`** and `m1_2.jpg` to **`image23.jpg`**
    - Rename `m113_1.jpg` to **`image12.jpg`** and `m113_2.jpg` to **`image27.jpg`**
    - Rename `m2_1.jpg` to **`image09.jpg`** and `m2_2.jpg` to **`image25.jpg`**
    - Rename `mi24_1.jpg` to **`image15.jpg`** and `mi24_2.jpg` to **`image30.jpg`**

- Rename `mi4_1.jpg` to **`image05.jpg`** and `mi4_2.jpg` to **`image20.jpg`**
- Rename `mi8_1.jpg` to **`image07.jpg`** and `mi8_2.jpg` to **`image24.jpg`**
- Rename `t64_1.jpg` to **`image11.jpg`** and `t64_2.jpg` to **`image29.jpg`**
- Rename `t72_1.jpg` to **`image06.jpg`** and `t72_2.jpg` to **`image22.jpg`**
- Rename `t80_1.jpg` to **`image01.jpg`** and `t80_2.jpg` to **`image17.jpg`**
- Rename `uh60_1.jpg` to **`image14.jpg`** and `uh60_2.jpg` to **`image21.jpg`**

At this point, your deployment1 folder should be set up as in Figure (All .jpg image files present in this folder are not shown in this Figure 31).



Contents of 'E:\threat_test\deployment1'

Name	Size
Answer_key.txt	0KB
image01.jpg	86KB
image02.jpg	37KB
image03.jpg	33KB
image04.jpg	57KB
image05.jpg	110KB
image06.jpg	62KB
image07.jpg	35KB
image08.jpg	48KB
image09.jpg	93KB
image10.jpg	44KB
image11.jpg	90KB
image12.jpg	78KB

Figure 31: deployment1 Folder Contents.

9. Navigate to the `deployment1` folder (you should still be in this folder from work done in step eight) and double click on the `Answerkey.txt` file to open it in the text editor on your computer. In this step you will enter the vehicles in the order you listed them in the step five above. This ordering must match the vehicle order and naming order in which the vehicles were listed in step six above. This does not equate to a display order for the images; it is how the system matches images with the correct response for grading purposes. Separate each vehicle with a press of the return or enter key. Be consistent in how you enter the names of the systems, i.e. all capital or lower case, with or without hyphens. How the answers are typed into the `Answerkey.txt` is how the user must type in his or her answer while using the program. In this case enter the vehicle names as follows (it does not matter whether you have a return after the last vehicle entry of the answer key text file):

```
t80
m1
brdm2
ah64
mi4
t72
mi8
btr70
m2
lav25
t64
m113
chieftain
uh60
mi24
chieftain
t80
lav25
brdm2
mi4
uh60
t72
```

m1
mi8
m2
btr70
m113
ah64
t64
mi24

Double check to ensure that you do not have any errors in the `Answerkey.txt`. Save your changes. Close the `Answerkey.txt` file. Rename `Answerkey.txt` to `deployment1.txt` (this is the file name variable that the program looks for when running).

This completes the steps necessary to develop your first threat identification test. By using the same steps above, with other weapon system images, you can develop tests specific to meet you unit needs.

3. Using the System

To use the system with the test version created in section one above follow the following steps.

1. Navigate to the `threat_test` folder on your hard drive.
2. Double click on `threat_test.exe` to start the application.
3. Enter you rank, first name, and last name in the initial login screen.
4. Enter `deployment1` when asked what test version you want to take.
5. Enter the desired time in seconds that you want each image displayed during the test. This also corresponds to the amount of time you will have to type in your answer. New users may want to enter 15-20 seconds until they are familiar with the interface of the

system. Users with more experience with the system should use the standard time allotted for their unit threat identification training.

6. After entering the time in step 5, the program will wait until you press the enter key or click the mouse button anywhere on the display screen. Once this is done, the program will immediately start displaying the images, one after another until all 30 images have been displayed.

7. After all images have been displayed, grading information will be displayed on the screen. An information text file will also be written to the hard drive in the threat_test folder. This text file will be named with whatever last name you entered in the login information of step 3.

8. Open this text file in any text editing or word processor application to view your testing information. Consecutive runs by a user with the same last name will add to the existing text file for that user.

D. CONSIDERATIONS WHEN DEVELOPING DIFFERENT TEST VERSIONS

1. Checklist

The following checklist is provided as a guide to help users develop additional tests with the MTTS testing subsystem. It assumes that you have previously completed the MTTS training subsystem tutorial.

√	STEP	
	1	Verify that the Hide MS-DOS file extensions for file types that are registered is unchecked.
	2	Create a folder on your hard drive named `threat_test`.
	3	Use the copy and paste command to copy the following files from the Tutorial folder on the CD-ROM into the threat_test folder created in step two: `threat_test.exe` (a single file), `Xtras` (a folder that contains four files).

	4	In the `threat_test` folder created in step two, create a sub folder named `deployment1`.
	5	Use the copy and paste command to copy the file `Answerkey.txt` (in the Tutorial folder) into the `deployment1` folder created in step four.
	6	Determine an order for the images that you are testing.
	7	Use the copy and paste command to copy the 30 JPEG images you are testing into the `deployment1` folder.
	8	Based on the order determined in step six above rename the thirty images in the `deployment1` folder. image01.jpg, image02.jpg…image30.jpg.
	9	Open the `Answerkey.txt` file and enter the image names in the order you listed them in step six above. This ordering must match the vehicle order and naming order in which the images were listed in step six. Separate each image with a press of the return or enter key. Be consistent on naming conventions, i.e. all capitals, no hyphens, etc. How the answers are entered into this file is how the user must type in their answers when taking the test.
	10	Check the `Answerkey.txt` file for any errors. Close the `Answerkey.txt` file.
	11	Rename `Answerkey.txt` to `deployment1.txt`
	12	Navigate to the `threat_test.exe` file on your hard drive
	13	Double click on `threat_test.exe` to run the program.

2. Troubleshooting

The following section covers some of the possible errors that can arise when using the MTTS testing subsystem. Possible solutions to the problem are also given.

a. Images Fail to Appear

If the JPEG images are named incorrectly, the program will not be able to import them into the system at run time. Double check to ensure that all JPEG images are correctly named. The JPEG images must be named as follows: `image01.jpg`, `image02.jpg`, `image03.jpg`, …`image10.jpg`, …`image30.jpg`.

All JPEG images must be in the subfolder created in step four of the checklist. If the images are not in this subfolder, the program will not find them during runtime. Ensure that all images are in this subfolder.

b. Grading issues

At the completion of a test run, the program will display the test score for that run. There are three main reasons why the program will give incorrect grading results.

The `Answerkey.txt` file is incorrectly named. This file must match the file name the user types in during the login sequence when prompted to enter what test version/number they wish to take. If these do not match, the program will not have an answer key to grade the test with. In this case, the program will score the test as zero out of zero. Along these lines, if the user types in the wrong test version, or mistypes the test version in during the login sequence, the results will be the same as a misnamed `Answerkey.txt` file. See the subsection on naming conventions for more information.

Another possible reason for incorrect grading is an error in what was listed as the correct answer for an image file in the `Answerkey.txt` file and what the image file really is. When developing test versions, it is important to ensure the correct ordering of images and corresponding answers in the `Answerkey.txt` file. For example, if `image03.jpg` is an image of an M1 tank, then the third entry in the `Answerkey.txt` file must correspond to this. Likewise, if `image27.jpg` is an M2 APC, then the 27th entry in the `Answerkey.txt` file must be M2 (or m2 if using lower case).

c. Authorware Error Message

An Authorware error message stating "unable to display icon image1 because the following error has occurred: Xtra not found" can occur when the program can not find the Xtra folder or the necessary files it contains.

Ensure that the Xtra folder is in the threat_test folder created in step two of the checklist. Ensure that the Xtra folder contains the four individual files `Jpegimp.x32, Mix32.x32, Mixview.x32, and Viewsvc.x32`. Using the copy and paste command to copy the entire folder from the Tutorial folder on the CDD-ROM into your folder should copy all files within the folder.

3. Naming conventions

When creating your own test versions, it is important to remember that certain names used in the checklist can change, and some can not.

a. Unchangeable Names

All image files must be named `image01.jpg` through `image30.jpg`. The program will not read in these files if they are named anything else. The Xtra folder cannot be renamed, nor can any of the files this folder contains.

b. Changeable Names

The folder created in step two, and subfolder created in step four, and the renamed `Answerkey.txt` file can all have different names. Ensure that the name for the subfolder and the renamed `Answerkey.txt` file are exactly the same. If the subfolder is named `Bosnia`, the answer key must be named `Bosnia.txt`.

APPENDIX D. IMPLEMENTATION CODE LISTINGS

This appendix presents examples of the Director 6.0 scripting language LINGO, and Authorware Attain 5.0 functions used in developing the MTTS.

A. LINGO COMMANDS

1. Navigation Within a Movie

```
on exitFrame
  go to the frame
end
```

2. Navigation to a Different Movie

```
on mouseUp
  go to movie "armor_training.dir"
end
```

B. AUTHORWARE FUNCTIONS

1. Capturing User Input

```
UserName:= EntryText
Rank:=GetWord(1, UserName)
FirstName:=GetWord(2, UserName)
LastName:=GetWord(3, UserName)
```

2. Reading in the Answer Key Text File

```
Data:= ReadExtFile(FileLocation^file^"\file"^".txt")
Records:=LineCount(Data)
```

3. Storing Text File as Array Elements

```
ReadData:=GetLine(Data,1)
ArraySet(Line, ReadData)
Line:=Line + 1
Data:=DeleteLine(Data, 1)
```

4. Grading

```
TotalNumber:=Records
repeat with I = 1 to Records
  if ArrayGet(I-1) = ValueAtIndex(AnswerArray, I) then
    NumberCorrect:=NumberCorrect + 1
  else
    questionsMissed:=questionsMissed^I^”,”
  end if
end repeat
```

LIST OF REFERENCES

[BIXL77] Bixler, B. and Bergman, T., Selecting and Implementing Computer-Based Training, [On-line]. Available at URL: http://www.clat.psu.edu.homes/bxb11/CBTGuide/CBTGuide.html

[DEVI97] Devin, P. D. and Robyn, A. E., Evaluation of the NJROTC Multimedia Instructional System, Santa Monica, California: RAND, 1997.

[DEVO95] Devore, J. L., Probability and Statistics for Engineering and the Sciences, Pacific Groove, CA: Brooks/Cole Publishing Company, 1995.

[GERY87] Gery, G., Making CBT Happen, Boston, Massachusetts: Weingarten Publications, Inc., 1987.

[HASS86] Hasset, J. and Dukes, S., The New Employee Trainer: A Floppy Disk, Psychology Today, September 1986.

[HAIL98] Hailey, D. E. and Hailey, C., Hypermedia, Multimedia, and Reader Cognition: An Emperical Study, Technical Communication, August 1998.

[HIXD93] Hix, D. and Hartson, H. R., Developing User Interfaces Ensuring Usability Through Product and Process, New York, New York: John Wiley and Sons, Inc., 1993.

[HORT94] Horton, W., Designing and Writing Online Documentation, New York, New York: John Wiley and Sons, 1994.

[JONA93] Jonassen, D. and Wang, S., Acquiring Structural Knowledge from Semantically Structured Hypertext, Journal of Computer-Based Instruction, Volume 20, 1993.

[KEAR83] Kearsley, G., Computer-Based Training, Reading, MA: Addison-Wesley, 1983.

[KEAR84] Kearsley, G., Training and Technology A Handbook for HRD Professionals, Reading, MA: Addison-Wesley, 1984.

[REEV93] Reeves, T., Pseudoscience in Computer-Based Instruction: The Case of Leamer Control Research, Journal of Computer-Based Instruction, Volume 20, 1993.

[SALO98] Salopek, J. J., Multimedia Based Training, Training and Development, November 1998.

[SEID85] Seidel R. J. and Weddle P. D., Computer-Based Instruction in Military Environments, New York, New York: Plenum Press, 1987.

[WALL97] Wallace, D. and Mutooni, P., A Comparative Evaluation of World Wide Web Based and Classroom Teaching, Journal of Engineering Education, 86(3), 1997.

[ZYDA97] Zyda, M. and Sheehean, J., Modeling and Simulation: Linking Entertainment & Defense, Washington, D.C.: National Academy Press, 1997.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Road, Ste 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5101

3. Chairman MOVES Academic Group 1
   Dr. Michael J. Zyda, Code CS/Zk
   Computer Science Department
   Naval Postgraduate School
   Monterey, CA 93943-5101

4. Academic Associate, Moves Academic Group 1
   Dr. Rudy Darken, Code CS/Dk
   Computer Science Department
   Naval Postgraduate School
   Monterey, CA 93943-5101

5. Dr. Geoffrey Xie, Code CS/Xg 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, CA 93943-5101

6. John S. Falby, Code CS/Fa 2
   Computer Science Department
   Naval Postgraduate School
   Monterey, CA 93943-5101

7. Dr. Don Brutzman, Code UW/Br 1
   Naval Postgraduate School
   Monterey, California 93943-5101

8. National Simulation Center (NSC) 1
   ATTN:ATZL-NSC (Jerry Ham)
   410 Kearney Avenue --- Building 45
   Ft. Leavenworth, KS 66027-1306

9. CPT Douglas S. Miller 3
   1460 Canoe Creek Drive
   Colorado Springs, CO 80906

10. Dr. Douglas R. Miller 1
HC01 Box 255
Marquette, MI 49855